

UNCLASSIFIED

AD 268 045

*Reproduced
by the*

**ARMED SERVICES TECHNICAL INFORMATION AGENCY
ARLINGTON HALL STATION
ARLINGTON 12, VIRGINIA**



UNCLASSIFIED

NOTICE: When government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related government procurement operation, the U. S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

ASD TECHNICAL REPORT 61-228

268 045 CATALOGED BY ASUA 268045
AS 61-228

MAGNETIC LOGICAL TRANSDUCERS

O. STRAM
S. EINHORN
J. CELIA

BURROUGHS CORPORATION
BURROUGHS LABORATORIES
RESEARCH CENTER
PAOLI, PA.

CONTRACT Nr AF 33(616)-6355

AUGUST 1961

ELECTRONIC TECHNOLOGY LABORATORY
AERONAUTICAL SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
WRIGHT-PATTERSON AIR FORCE BASE, OHIO

NOTICES

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Qualified requesters may obtain copies of this report from the Armed Services Technical Information Agency, (ASTIA), Arlington Hall Station, Arlington 12, Virginia.

This report has been released to the Office of Technical Services, U. S. Department of Commerce, Washington 25, D. C., for sale to the general public.

Copies of ASD Technical Reports and Technical Notes should not be returned to the Aeronautical Systems Division unless return is required by security considerations, contractual obligations, or notice on a specific document.

MAGNETIC LOGICAL TRANSDUCERS

*O. STRAM
S. EINHORN
J. CELIA*

*BURROUGHS CORPORATION
BURROUGHS LABORATORIES
RESEARCH CENTER
PAOLI, PA.*

AUGUST 1961

ELECTRONIC TECHNOLOGY LABORATORY

*CONTRACT Nr AF 33(616)-6355
PROJECT Nr 7062
TASK Nr 70949*

*AERONAUTICAL SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
WRIGHT-PATTERSON AIR FORCE BASE, OHIO*

FOREWORD

This report was prepared by Research Center, Paoli, Pennsylvania, on Air Force contract AF33(616)6355, under Task Nr 70949 of Project Nr 7062, "Magnetic Logical Transducers." The work was administered under the direction of Electronic Technology Laboratory, Aeronautical Systems Division, Wright-Patterson Air Force Base, Ohio. Mr. Eugene C. Maupin was task engineer for the laboratory.

The studies presented began in June 1959 and were concluded in May 1961. Although the studies were a group effort the chief contributors were O. Stram, J. Celia, and S. Einhorn.

This report concludes the work on contract AF 33(616)6355.

ABSTRACT

A method is presented for the logical design of single-stage, combinatorial switching circuits of n -variables. This method is applicable to circuits composed of threshold devices, such as magnetic cores, transistors with Kirchhoff adder inputs, parametrons, etc. A study of the constraints imposed by the form of the input portions of the threshold devices leads to the definition of certain classes of functions which are physically realizable in a single device. By the use of this method, arbitrary switching functions of as many as seven variables have been easily designed by hand computations.

An algorithm for mechanizing Boolean switching functions, by means of a net of magnetic toroidal cores, is described. The algorithm is referred to as Simplex which, in this application, is programmed for a digital computer. Computer-derived solutions specify the wiring configuration for a core or net of cores yielding a device for performing combinatorial logic. Switching functions are realizable in essentially one clock time. The logical designer may synthesize a function directly from the truth table without proceeding in the customary manner of expressing the function of Boolean canonical form and then attempting to minimize with respect to hardware or other criteria by means of algebraic manipulation or some mapping or charting technique.

An updated bibliography is included.

PUBLICATION REVIEW

The publication of this report does not constitute approval by the Air Force of the findings or conclusions contained herein. It is published only for the exchange and stimulation of ideas.

FOR THE COMMANDER:



LESLIE E. KNAPP, Capt, USAF
Acting Chief, Bionics & Computer Branch
Electronic Technology Laboratory

Signature
Title
Laboratory

TABLE OF CONTENTS

	Page
SECTION I - ARBITRARY BOOLEAN FUNCTIONS ON N-VARIABLES REALIZABLE IN TERMS OF THRESHOLD DEVICES	
Introduction	1
Threshold Device Logical Element	3
Symmetry Operations	5
Function Transformation	9
Consistent Functions	13
Non-Consistent Functions	22
The Partitioning of Functions	23
Classification of Realizable Functions	27
 SECTION II - THE USE OF THE SIMPLEX ALGORITHM IN THE MECHANIZATION OF BOOLEAN SWITCHING FUNCTIONS BY MEANS OF MAGNETIC CORES	
Introduction	35
Notation	36
The Voting Function	37
The Simplex Algorithm	40
Explanation of Simplex	44
Simplex Modifications	56
Variable Suppression	58
Tableau Structure	62
Computational Procedure	69
Don't Care Condition	76
 SECTION III - CONCLUSIONS	77
 SECTION IV - RECOMMENDATIONS	
Theoretical Studies	79
Simplex	79
 BIBLIOGRAPHY	81
 GLOSSARY	91

LIST OF ILLUSTRATIONS

			Page
FIGURE	1a	Non Realizable Functions	12
	1b	Partitioned Functions	12
FIGURE	2	A Portion of a Truth Table for N-Variables	15
FIGURE	3	Function Profiles	17
FIGURE	4	Sum of Minimal Profiles	19
FIGURE	5	A Consistent Three-Variable Function and Solution	20
FIGURE	6	Consistent Minimal Profiles	20
FIGURE	7	Circle of Compatibility	21
FIGURE	8	Hypothetical EXCLUSIVE-OR Profile	22
FIGURE	9a	Profile Shift Criterion for Realizability	28
	9b	Profile Shift Criterion for Realizability	29
FIGURE	10	Mirror Symbol Notation for Core Winding	37
FIGURE	11	Truth Table for Voting Function where $\mu = 2$ and $n = 3$	38
FIGURE	12	Core and Wiring Configuration for Voting Function where $\mu = 2$ and $n = 3$	38
FIGURE	13	Truth Table for $f = 01011101$	40
FIGURE	14	Two-Core Mechanization of $h = 011000010$	43
FIGURE	15	Convex Polygon of Equations (21)	44
FIGURE	16	Corner Points of the Convex Polygon of Equations (22)	46
FIGURE	17	Truth Table for f and the Sub Functions	59
FIGURE	18	Cores and Winding Arrangement for Mechanizing f	61

LIST OF TABLES

TABLE	I	Consistent Solutions of Two Variable Functions	16
TABLE	II	Partitioning of a Five Variable Function	25
TABLE	III	Primitive Solutions of Consistent Four-Variable Functions	31
TABLE	IV	Primitive Functions of up to Five Variables	33

SECTION I

ARBITRARY BOOLEAN FUNCTIONS ON N-VARIABLES REALIZABLE IN TERMS OF THRESHOLD DEVICES

INTRODUCTION

It is well known that there are 2^{2^n} possible switching functions of n-variables, where each variable can take only the values "1" or "0." Each truth function has 2^n states. For the purpose of designing arbitrary functions, switching circuits for only a small class of these functions need to be catalogued, since it can be shown that this class of circuits, under the symmetry operations of permutations and complementations of the input variables, maps the entire set.

Aiken⁵¹ lists all vacuum tube switching circuits for four variables in a table of 402 entries. Two other tables are given which show the transformation of the function under the group of symmetry operations on the input variables, but even for four variables, the process of transforming and classifying an arbitrary function is extremely tedious, while five variables require a catalog of over one million switching circuits.⁵³

The above classification is without regard to the properties of the physical circuit which is used to embody the function. Thus, the same "primitive" set of functions is designed whether the circuit elements are vacuum tubes, magnetic cores, or relays, etc. If the circuit element is a threshold device such as a magnetic core, then certain constraints are introduced into the problem. These constraints offer a different class of functions, those which can be embodied in a

Manuscript released by the authors (May 25, 1961) for publication as an ASD Technical Report.

single input-composite of a threshold device. Such functions will be called consistent with respect to a threshold device. If only the consistent switching circuits need to be designed and allowing symmetry operations, a table of only 14 entries is required for all four variable functions, and probably no more than 65 for all five variable functions.

The design of an arbitrary n-variable function, using threshold devices is accomplished in the following manner: a) The truth function is first partitioned so as to separate the "one" bits into a set of functions, each of which is independently consistent. b) The input composites for each of these functions is designed, perhaps by referring to a table of consistent switching circuits. c) Since the partitioning of the function places each of its "ones" into one, and only one, sub-set, the output of the switching circuits need only to be ORed together to embody the entire function.

It is the purpose of this report to demonstrate a means of partitioning and designing arbitrary n-variable Boolean functions. Although most of the reported work is with reference to magnetic cores, it is presented in terms of the generalized concept of a threshold switching device. For this reason and because they are adequately treated elsewhere,⁵⁶ the physical details of magnetic core switching is omitted here.

This report also covers a computer technique for mechanizing any arbitrary Boolean function with toroidal cores. In other words, the computer-derived solution will indicate the wiring configuration for a core or net of cores yielding a device for performing combinational logic. Switching functions of n-variables are realizable in one clock time.

The technique is a linear programming technique referred to as the Simplex algorithm. Of extreme importance is the fact that the logical designer may synthesize a function directly from the truth table without proceeding in the customary manner of expressing the function in Boolean canonical form and then attempting to minimize the hardware by algebraic manipulation or some other charting or mapping technique.

THRESHOLD DEVICE LOGICAL ELEMENT

A threshold device can be brought to either of two distinct states depending upon the magnitude of some physical quantity at its input. Although the threshold of a given device has a distinct magnitude (i.e. in a magnetic core the applied magnetizing force, H_i , must exceed the coercive force, H_c , in order to switch) it is preferable to specify a range. That is if the state is not to change, the strength of the input must be less than a specified quantity; if it is to change, the strength of the input must be greater than another quantity. To use again the example of a magnetic core, if the core is not to switch from negative remanence, the applied magnetizing force $H_i \leq 0$; or if it is to switch $H_i \geq H_o$, where $H_o \geq H_c$ and depends upon the desired speed of switching. In units of H_o : $H_i \leq 0$ implies a "0," $H_i \geq 1$ implies a "1," where "0" and "1" denote respectively the negative and positive remanent states of the core and it is assumed that the core always starts from negative remanence.

The threshold device may be considered to be composed of two separate portions, an input circuit which comprises a means of weighting and combining the input variables, and an output stage which indicates by a definite change in

state whether or not the combined value of the weighted variables exceeds the threshold. Following Karnaugh⁵⁶ the mathematical description of the input circuit is called the input composite. The input variables are denoted by $x_1 x_2 \dots x_n$ and can take only the values 0 or 1. All variables operate simultaneously in the input composite. The term x_0 is reserved for a unit pulse, which together with its weighting factor, is called the bias.

The effects of the inputs are added algebraically to obtain the net effect on the output stage of the threshold device, and the input composite may, therefore, be represented in the following form:

$$M = q_0 N_0 x_0 + q_1 N_1 x_1 + \dots + q_n N_n x_n \quad (1)$$

Here $q_i = \pm 1$ and N_i is a positive integer or zero. In terms of magnetic cores, q_i represents the polarity, N_i the number of turns of the winding, and M the net applied magnetomotive force where x_i represents a constant current pulse.

Now, since the value of M depends on the manner in which the variables x_i are taken as 0 and 1, we define a new quantity in terms of a column matrix which will be called the solution vector, S, the terms of which are called the input values.

$$\underline{S} = \begin{bmatrix} q_0 N_0 \\ q_1 N_1 \\ \vdots \\ q_n N_n \end{bmatrix}$$

The M above is one term of a vector which is found by multiplying the solution vector by the truth table for n -variables, treated as a matrix T. The vector M will be called the input composite vector.

$$\underline{T} \cdot \underline{S} = \underline{M}$$

$$x_0 \ x_1 \ x_2 \ x_3 \ \dots \ x_n$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & \dots & 0 \\ 1 & . & . & . & \dots & . \\ 1 & . & . & . & \dots & . \\ . & . & . & . & \dots & . \\ . & . & . & . & \dots & . \\ . & . & . & . & \dots & . \\ 1 & 1 & 1 & 1 & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} q_0 N_0 \\ q_1 N_1 \\ q_2 N_2 \\ . \\ . \\ . \\ q_n N_n \end{bmatrix} = \begin{bmatrix} q_0 N_0 \\ q_0 N_0 + q_1 N_1 \\ q_0 N_0 + q_1 N_1 + q_2 N_2 \\ q_0 N_0 + q_1 N_1 + q_2 N_2 \\ q_0 N_0 + q_1 N_1 + q_2 N_2 + q_3 N_3 \\ . \\ . \\ . \\ q_0 N_0 + q_1 N_1 + q_2 N_2 + q_3 N_3 + \dots + q_n N_n \end{bmatrix}$$

The logic function F is evaluated by applying the threshold condition to M.

SYMMETRY OPERATIONS

The input circuit of a threshold device is specified by its solution vector. Since it is a physical structure, it is invariant to complementations of and permutations among the input variables. It is easily shown that the effect of the symmetry operations is to permute the rows of the truth table. Consequently, the terms of the input composite vector are likewise permuted without changing their values. One can, thus, postulate a $2^n \times 2^n$ permutation-matrix P which transforms a given M or its corresponding logic function, F, to another F' having the same primitive solution, or switching circuit. Now, since it is desirable to avoid the use of complement drivers, and the necessity for interchanging leads, we transform the solution matrix, S, by means of a matrix, C, which is defined as follows:

$$\underline{T} \cdot \underline{S} = \underline{M} \quad (2)$$

$$\underline{T} \cdot \underline{C} \cdot \underline{S} = \underline{M}' = \underline{P} \cdot \underline{T} \cdot \underline{S} = \underline{P} \cdot \underline{M} \quad (3)$$

$$\therefore \underline{T} \cdot \underline{C} = \underline{P} \cdot \underline{T} \quad (4)$$

pre-multiplying both sides by \underline{T}_t

$$\underline{T}_t \cdot \underline{T} \cdot \underline{C} = \underline{T}_t \cdot \underline{P} \cdot \underline{T} \quad (5)$$

It can be shown by partitioning \underline{T} and applying induction that $\underline{T}_t \cdot \underline{T}$ is non-singular and thus has an inverse, $(\underline{T}_t \cdot \underline{T})^{-1}$ for any number of variables, n.

$$\underline{C} = (\underline{T}_t \cdot \underline{T})^{-1} \cdot \underline{T}_t \cdot \underline{P} \cdot \underline{T} \quad (6)$$

Only certain \underline{P} -matrices are allowed because, although the transformation, $\underline{M}' = \underline{P} \cdot \underline{M}$, is not unique, not all of the possible \underline{P} -matrices will lead to a product, $\underline{T}_t \cdot \underline{P} \cdot \underline{T}$, which is non-singular. The properties of the allowed \underline{P} -matrix have not been fully investigated as yet. In any case, we are now in a position to describe \underline{C} -matrices which will perform the symmetry operations, on both the solution \underline{S} and the truth matrix \underline{T} , by (4) above. An example will, perhaps, best bring out the qualities of the \underline{C} -matrix. Given a Boolean function of 3 variables, $F = 101011\underline{00}$, (the least significant bit is underlined), the permutation matrix, \underline{P} , below transforms it to $F' = 0010011\underline{1}$ which has the same switching circuit and which is listed in a table of basic switching circuits such as that of Aiken.⁵¹

$$\underline{F}' = \underline{P} \cdot \underline{F} \quad (7)$$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

The C matrix is formed by the use of formula (6)

$$\underline{C} = (\underline{T}_t \cdot \underline{T})^{-1} \cdot (\underline{T}_t \cdot \underline{P} \cdot \underline{T})$$

For $n = 3$, the product $\underline{T}_t \cdot \underline{T}$ and its inverse are:

$$\underline{T}_t \cdot \underline{T} = \begin{bmatrix} 8 & 4 & 4 & 4 \\ 4 & 4 & 2 & 2 \\ 4 & 2 & 4 & 2 \\ 4 & 2 & 2 & 4 \end{bmatrix} \quad (\underline{T}_t \cdot \underline{T})^{-1} = \frac{1}{4} \begin{bmatrix} 2 & -1 & -1 & -1 \\ -1 & 2 & 0 & 0 \\ -1 & 0 & 2 & 0 \\ -1 & 0 & 0 & 2 \end{bmatrix}$$

Then $\underline{T}_t \cdot \underline{P} \cdot \underline{T}$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 8 & 4 & 4 & 4 \\ 4 & 2 & 2 & 0 \\ 4 & 2 & 0 & 2 \\ 4 & 0 & 2 & 2 \end{bmatrix} \quad (8)$$

$$(\underline{T}_t \cdot \underline{T})^{-1} \cdot (\underline{T}_t \cdot \underline{P} \cdot \underline{T}) = \underline{C}$$

$$\frac{1}{4} \begin{bmatrix} 2 & -1 & -1 & -1 \\ -1 & 2 & 0 & 0 \\ -1 & 0 & 2 & 0 \\ -1 & 0 & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 8 & 4 & 4 & 4 \\ 4 & 2 & 2 & 0 \\ 4 & 2 & 0 & 2 \\ 4 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \quad (9)$$

Since post-multiplying \underline{T} by \underline{C} is the same as pre-multiplying it by \underline{P} , let us form the product $\underline{T} \cdot \underline{C} = \underline{T}'$ and examine the resulting transformation of the truth matrix.

$$\begin{array}{c} \underline{T} \\ x_0 \ x_1 \ x_2 \ x_3 \end{array} \cdot \underline{C} = \underline{T}'
\quad
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (10)$$

Comparing \underline{T} and \underline{T}' , we see that the effect of the matrix \underline{C} has been to interchange columns x_1 and x_3 and replace the variables by their complements x_1' , x_2' , x_3' . Indeed, \underline{C} is the product of a permutation matrix $\underline{\pi}$ and a complementation matrix \underline{K} .

$$\underline{\pi} \cdot \underline{K} = \underline{C}
\quad
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \quad (11)$$

Thus, to complement any variable x_i without recourse to complement drivers, it is replaced by $(x_0 - x_i)$, this corresponds to Karnaugh's theorem.⁵⁶

All possible matrices $\underline{\pi}$ and \underline{K} for n -variables, together with the unit matrix \underline{I} , form a group which is isomorphic with the group of isometries of an n -dimensional hypercube. For $n = 3$, the group contains 48 members, i.e., six permutations, and eight complementations.

We have, thus, shown that if a permutation matrix \underline{P} can be found which transforms an arbitrary function to another having a known solution (or switching circuit) and if \underline{P} has the property such that $\underline{T}_t \cdot \underline{P} \cdot \underline{T}$ is non-singular, then a transformation \underline{C} can be found which will convert the known solution to the desired one.

FUNCTION TRANSFORMATION

A function will be called in basic form when it has been transformed by a set of symmetry operations to another function having the same physical switching circuit, but such that all the true bits have been moved as closely as possible to the least significant bit end of the function. The required set of symmetry operations can be selected in the following manner.

Consider a truth function on n -variables. The n th variable is false for the first half and true for the second half. Count the number of true states of the function in the second half of the function and subtract from the number of true bits in the first half of the function. Call this number Δ_n . If $\Delta_n < 0$, there are more true states in the second half of the function than in the first half. Thus, complementing the variable X_n is one of the required symmetry operations. A set of numbers Δ_i , $1 \leq i \leq n$, is found by subtracting the number of states of the function which are true when the variable, X_i is true from the number of states which are true when the variable is false. With this set of numbers, it is possible to describe the required set of symmetry operation.

Consider now a function in basic form which has k true states followed by $(m - k)$ false states where $m = 2^n$. The following statements concerning the Δ and solution vectors can be inferred from the form of the function. Proofs are omitted here.

- 1) Parity is conserved for all k among k and Δ_i . This is also true for any function.

2) The Δ_i can have the following values only:

$$0 \leq \Delta_1 \leq 1$$

$$0 \leq \Delta_2 \leq 2$$

$$0 \leq \Delta_3 \leq 4$$

$$0 \leq \Delta_4 \leq 8$$

$$\vdots \quad \vdots \quad \vdots$$

$$0 \leq \Delta_n < 2^{n-1}$$

$$3) \Delta_{j-1} \leq \Delta_j$$

4) All k true out of m functions with no gaps are realizable.

5) The minimal solution vector will have the following form:

$$q_0 = +1$$

$$q_i = -1 \quad \text{for} \quad 1 \leq i \leq n$$

$$N_1 \leq N_2 \leq \dots \leq N_n \leq N_0$$

6) The statements are equivalent that the given function is in basic form when a) the Δ_i are in monotonically increasing order or b) the solution if it exists is as defined in 5 above.

We shall now state by analogy with the function studied that any arbitrary function is in basic form when its Δ_i are all positive and in monotonically increasing order as i increases. The set of symmetry operations required to transform the arbitrary function to basic form is identical with the operations required to transform the set Δ_i to basic form. Since complementations and permutations do not in general commute, all variables with negative Δ_i are first complemented and then the indicated permutations are performed.

With some practice, the transformation of the functions for any symmetry operation becomes obvious. For example, interchanging x_1 and x_2 permutes the terms of the pairs: 1,2; 5,6; 9,10; 13,14; etc. of the truth table. Complementing x_1 permutes the terms of the pairs: 0,1; 2,3; 4,5; etc.

The \underline{C} matrix required to transform the solution in basic form, \underline{S}' , to the required solution, \underline{S} , is found as follows. If $\underline{\Delta}$ is the vector for the given function and $\underline{\Delta}'$ is the vector for the same function in basic form one can write a matrix \underline{D} such that

$$\underline{D} \cdot \underline{\Delta}' = \underline{\Delta}$$

Since the solution has the order $n+1$, the \underline{D} matrix is converted to the \underline{C} matrix by bordering with a row and a column. For example:

Example 1

$$\begin{aligned} F &= 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0; & \Delta_1 &= -3 & \Delta_2 &= 1 & \Delta_3 &= 1 \\ F' &= 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1; & \Delta'_1 &= 1 & \Delta'_2 &= 1 & \Delta'_3 &= 3 \end{aligned}$$

$$\underline{D} \cdot \underline{\Delta}' = \underline{\Delta}$$

$$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 3 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

$$\underline{C} = \begin{bmatrix} 1 & 0 & 0 & 1^* \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

* To perform complementation, every column which contains a negative entry must have a one in the added row.

$$\begin{bmatrix} \underline{S} \\ 0 \\ 2 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} \underline{C} \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \underline{S'} \\ 2 \\ -1 \\ -1 \\ -2 \end{bmatrix}$$

It is apparently true that the set, Δ_i , uniquely specifies the function if it is realizable in a single input composite, and it is obvious from the way they are defined that the absolute values of the members of the set for a given function are invariant to symmetry operations on the function, although their order may be interchanged. If the function is not realizable in a single input composite, the set, Δ_i , does not uniquely specify it, for example see figure 1a.

		A		B		
A	B	F _A	G _A	F _B	G _B	H _B
1	1	1	0	1	0	0
1	1	1	0	1	0	0
1	1	1	0	1	0	0
1	1	1	0	1	0	0
—	—	—	—	—	—	—
1	0	1	0	0	0	0
0	1	0	0	0	1	0
0	1	0	0	0	0	1
1	0	0	1	0	0	0
—	—	—	—	—	—	—
1	1	1	0	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
—	—	—	—	—	—	—
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

$$\begin{bmatrix} \underline{\Delta_A} \\ 1 \\ 1 \\ 3 \\ 5 \end{bmatrix} \quad \begin{bmatrix} \underline{\Delta_B} \\ 1 \\ 1 \\ 3 \\ 5 \end{bmatrix} \quad \begin{bmatrix} \underline{\Delta_{FA}} \\ 2 \\ 2 \\ 4 \\ 4 \end{bmatrix} + \begin{bmatrix} \underline{\Delta_{GA}} \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} \underline{\Delta_A} \\ 1 \\ 1 \\ 3 \\ 5 \end{bmatrix} ; \quad \begin{bmatrix} \underline{\Delta_{FB}} \\ 1 \\ 1 \\ 5 \\ 3 \end{bmatrix} + \begin{bmatrix} \underline{\Delta_{GB}} \\ -1 \\ 1 \\ -1 \\ 1 \end{bmatrix} + \begin{bmatrix} \underline{\Delta_{HB}} \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} \underline{\Delta_B} \\ 1 \\ 1 \\ 3 \\ 5 \end{bmatrix}$$

Figure 1a.
Non Realizable Functions

Figure 1b.
Partitioned Functions

The non-realizable functions A and B in figure 1a have the same set, Δ_1 , but neither can be transformed into the other by any set of symmetry operations, showing that the Δ_1 does not specify uniquely a non-realizable function. Partitioning the functions A and B in figure 1b it is seen that summing over the like terms of D vectors of the realizable functions making up the partitions produces the Δ vector of the non-realizable function.

Given a realizable function, if $\Delta_a = \Delta_b$ permuting the variables X_a and X_b will cause certain true states of the function to commute without changing the function, since it implies that $q_a N_a = q_b N_b$. If the function is not realizable, however, permuting the inputs X_a and X_b may change the function even if $\Delta_a = \Delta_b$. Thus, if the function is not realizable, there is no guarantee that ordering its Δ_1 monotonically will cause the true states to be clustered most densely. Nevertheless, the function will be considered in basic form when its set of Δ_1 is monotonically ordered w. r. t. increasing i.

CONSISTENT FUNCTIONS

If we examine a truth table for n-variables, we see that each span of four rows repeats the truth values for the variables x_1 and x_2 . The variable x_3 becomes true in the second span and remains true for the entire span, likewise with x_4 , in the third span. In fact, each variable, x_i , $3 \leq i \leq n$, must be constant in any arbitrary span of four rows. Thus, any variations in the values of the terms of the input composite vector within any span of four terms must be due to the variables x_1 and x_2 . Within any span, the presence of true variables

whose indices are 3 or greater only affect the net bias and establish an average level for that span as can be seen in figure 2. Consequently, it is instructive to study the properties of the set of functions of two variables.

Of the 16 functions of two variables, 14 of them are consistent as defined in the introduction. With a little reflection, it is possible to write unique, minimal solutions for each of them. These functions are shown in Table I, where the decimal heading indicates the function, i.e. one is 0001, five is 0101, etc. Six and nine, the two alternating symmetric functions (EXCLUSIVE OR and its inverse), cannot be embodied in a single input composite and are, therefore, inconsistent. It is easy to see that this is true by referring to figure 2. That is, for function "six," if both $q_1 N_1 + q_0 N_0 \geq 1$, and $q_0 N_0 + q_2 N_2 \geq 1$ where $q_0 N_0 \leq 0$, then surely $q_0 N_0 + q_1 N_1 + q_2 N_2 \geq 1$, and similarly for "nine."

As an example of the manner in which solutions of higher order functions can be built up from the two-variable solutions, compare the solution for function "seven" with the solution for function "one." We see that the two are identical if the x_0 input value of the former is reduced by one. If we take the three-variable function $f_3 = 0001, 0111$, the first span can be represented by function "seven" and the second by function "one." Referring to figure 2, in the second span, the net bias term, which is constant throughout the span is $q_0 N_0 + q_3 N_3$. If we, therefore, make the input value of x_3 , $q_3 N_3 = -1$, then the solution for the first span fits the second span and the input composite is $M = 2x_0 - 1x_1 - 1x_2 - 1x_3$. Plotting the successive values of M (the terms of the input composite vector) we get the input composite profile which aids in visualizing the described process. (See figure 3) Thus, the profiles of the minimal solutions of the functions "seven"

	x_0	x_1	x_2	x_3	x_4	\dots	x_n	M
Span 1	1	0	0	0	0	\dots	0	$q_0 N_0 (= M_0)$
	1	1	0	0	0	\dots	0	$q_0 N_0 + q_1 N_1$
	1	0	1	0	0	\dots	0	$q_0 N_0 + q_2 N_2$
	1	1	1	0	0	\dots	0	$q_0 N_0 + q_1 N_1 + q_2 N_2$
Span 2	1	0	0	1	0	\dots	0	$(q_0 N_0 + q_3 N_3)$
	1	1	0	1	0	\dots	0	$(q_0 N_0 + q_3 N_3) + q_1 N_1$
	1	0	1	1	0	\dots	0	$(q_0 N_0 + q_3 N_3) + q_2 N_2$
	1	1	1	1	0	\dots	0	$(q_0 N_0 + q_3 N_3) + q_1 N_1 + q_2 N_2$
Span 3	1	0	0	0	1	\dots	0	$(q_0 N_0 + q_4 N_4)$
	1	1	0	0	1	\dots	0	$(q_0 N_0 + q_4 N_4) + q_1 N_1$
	1	0	1	0	1	\dots	0	$(q_0 N_0 + q_4 N_4) + q_2 N_2$
	1	1	1	0	1	\dots	0	$(q_0 N_0 + q_4 N_4) + q_1 N_1 + q_2 N_2$
Span 2^n

	1	0	0	1	1	\dots	1	$(q_0 N_0 + q_n N_n + q_{n-1} N_{n-1} + \dots + q_3 N_3)$
	1	1	0	1	1	\dots	1	$(q_0 N_0 + q_n N_n + q_{n-1} N_{n-1} + \dots + q_3 N_3) + q_1 N_1$
Span 2^n	1	0	1	1	1	\dots	1	$(q_0 N_0 + q_n N_n + q_{n-1} N_{n-1} + \dots + q_3 N_3) + q_2 N_2$
	1	1	1	1	1	\dots	1	$(q_0 N_0 + q_n N_n + q_{n-1} N_{n-1} + \dots + q_3 N_3) + q_1 N_1 + q_2 N_2$

FIGURE 2 - A Portion of a Truth Table for
N-Variables

No.	0	1	2	3	4	5	7	8	10	11	12	13	14	15
Fn.	0	1	0	1	0	1	1	0	0	1	0	1	0	1
	0	0	1	1	0	0	1	0	1	1	0	0	1	1
	0	0	0	0	1	1	1	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	1	1	1	1	1	1	1
S.	0	1	0	1	0	1	2	-1	0	1	0	1	0	1
	0	-1	1	0	-1	-1	-1	1	1	1	0	-1	1	0
	0	-1	-1	-1	1	0	-1	1	0	-1	1	1	1	0

TABLE I. Consistent Solutions of Two Variable Functions

and "one" are identical in form, but that the latter is shifted one unit to the left. Likewise, the two spans of the function F are identical in form with the lower span shifted one unit to the left with respect to the upper span, in accordance with the input value of x_3 . The function F is, therefore, consistent.

Functions such as "seven" and "one" will be called compatible with one another if both functions can be derived from the same solution merely by a change in the bias. It can be shown that other compatibilities exist if certain equivalent modified two-variable solutions are admitted. The only allowed modifications of the minimal solutions are in the terms N_1 and N_2 of the solution vector. The signs of the terms, q_i , are not allowed to change. The N_i can only be increased, since we start with a minimal solution and reducing the input value of either variable to zero reduces the order of the solution. Likewise, if the input-value of x_i is zero, we cannot alter it without more information,

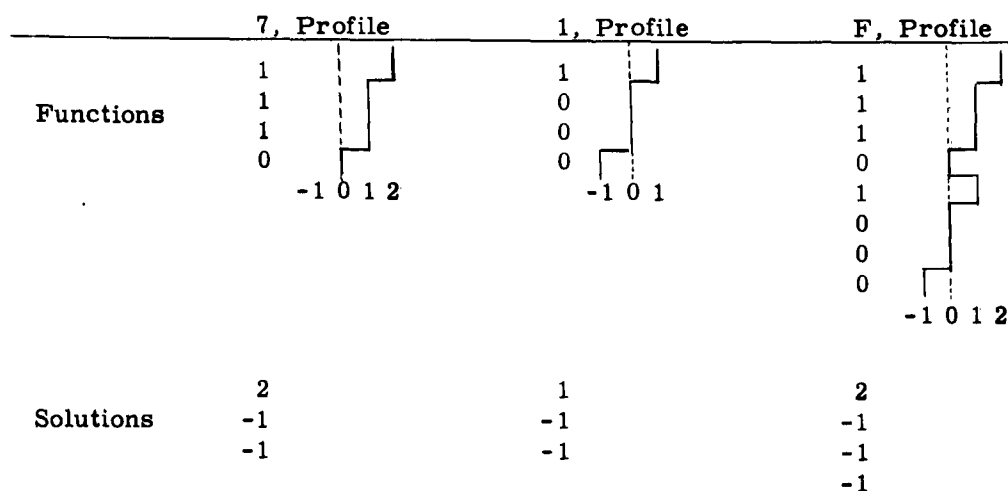


FIGURE 3 - Function Profiles

since the value of q_1 is not known. We now investigate whether a modified, or non-minimal solution, which may represent a different function can be restored to representing the original function merely by a change in bias.

If a modified two-variable solution is formed from a minimal two-variable solution by increasing the input value N_1 of one of the two variables, it can be seen by the first span of figure 2 that the value of the input composite for either pair of terms M_1 and M_3 or M_2 and M_3 will be altered, without altering the difference between them because the same amount is added to each. Thus, if the truth values of either or both members of one pair of terms change, their truth values can be restored by a compensating change in the bias. This compensating change will not alter the truth values of the unaffected terms for the following reasons: Consider first the terms M_1 and M_2 .

- (a) If both terms are greater than one, and if the input value of x_1 or x_2

is altered so as to cause either M_1 or M_2 to become zero or less (the q_i of the variable in question must be -1), then in "restoring" by a bias change, which must be in the positive direction, the previously unaffected term is made more positive so its truth value is unchanged.

(b) If both terms were zero or less, by the same reasoning, the truth value of the unaltered term is not affected by restoring.

(c) If one term is one or greater and the other is zero or less, the truth values of terms M_1 and M_2 cannot be altered by altering N_1 or N_2 since the signs q_1 and q_2 are opposite. Note that if either N_1 or N_2 is zero, it cannot be altered since we do not know the sign of its q_i .

Now consider the term M_0 , the truth value of which is most likely to change with restoring by bias compensation since it depends only on N_0 :

(a) If $M_0 \leq 0$ and the solution is to be altered so as to cause either M_1 or M_2 to exceed the threshold, it can only be done if the appropriate input value was initially positive. Restoring will, therefore, only make M_0 more negative and will not alter its truth value, f_0 .

(b) If $M_0 \leq 0$ and either M_1 or M_2 is to be made less than zero, it cannot be done by the allowed alterations, since the appropriate q_i must be initially positive.

(c) If $M_0 \geq 1$ the same reasoning shows that the only possible alteration in the term M_1 or M_2 is such as to cause its value to go from one or greater to zero or less, changing its truth value from a "1" to a "0." Therefore, restoring the truth value of the term M_1 or M_2 will not alter the truth value of M_0 .

Thus, it has been shown that a minimal two-variable solution which represents one two-variable function can be altered so as to represent a different, though

related, function by altering the input value N_i of one or both of the variables, x_1 , x_2 , or the bias x_0 and then "restored" to representing the original function solely by a change in the bias input value $q_0 N_0$. These related functions, sharing a non-minimal solution are also compatible.

As an example, let us solve the three-variable function $\underline{F} = 00010101$. Dividing it into two four-bit spans, noting their minimal, two-variable solutions from Table I, and generating their profiles, we have figure 4.

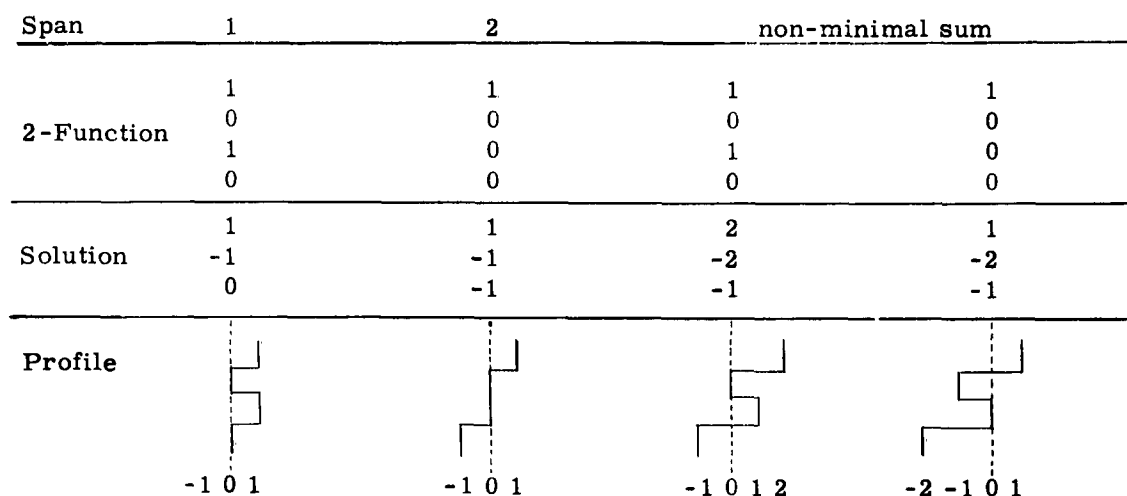


FIGURE 4 - Sum of Minimal Profiles

It is readily seen that no possible alteration of the input values of the biases of either two-variable solutions in figure 4 can make either function fit the other function. Now, if we add like terms of the solution vectors, or add across the two profiles, we obtain the non-minimal solution and profile shown in the third column of figure 4, which still represents the upper span function. The fourth column shows that by adding -1 to the bias, the same solution and profile can

also represent the second span of the function \underline{F} . This indicates, as in the previous example, that the input value of variable x_3 , $q_3 N_3 = -1$ and the solution for the function F is as shown in figure 5. If the two corresponding terms of two minimal solutions have opposite signs, the order of the solution is reduced and the sum cannot represent either. Such solutions are not compatible.

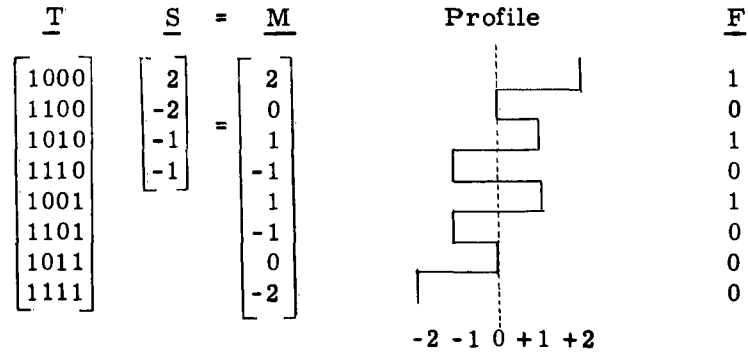


FIGURE 5 - A consistent Three-Variable Function and Solution

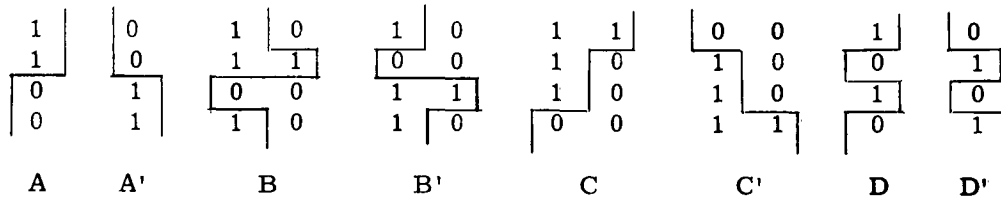


FIGURE 6 - Consistent Minimal Profiles

The fourteen consistent, two-variable functions may now be tabulated and classified with respect to their compatibility with one another. A convenient way to do so is on the basis of their profiles, figure 6.

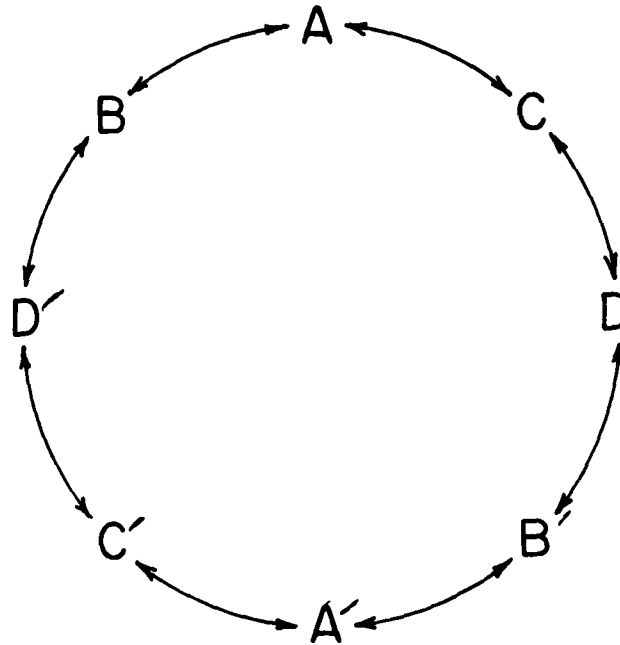


Figure 7 - CIRCLE OF COMPATIBILITY
EACH SOLUTION IS COMPATIBLE ONLY
WITH ITS NEAREST NEIGHBORS.

The function is written between the values $M = 0$ and $M = 1$. Where the same profile can represent two functions by shifting, they are displayed accordingly. The letters are arbitrarily assigned for convenient reference. The compatibilities of the minimal functions are shown in figure 7. The empty set is called "0," the filled set I and they are compatible with all other functions.

NON-CONSISTENT FUNCTIONS

It will be recalled that the functions "six" and "nine" the EXCLUSIVE-OR and its inverse are not consistent. A two-variable function can be thought of as being composed of four functions of $(n-2) = 0$ variables, and the profile of a zero-variable function is obviously a straight line at zero, or at one. Let us construct a hypothetical profile for the function "six" by shifting the zero variable profile to accommodate its four quarters, see figure 8.

First quarter		0	-	shift = +1
Second quarter		1		
Third quarter		1		- shift = -1
Fourth quarter		0		

Figure 8 - Hypothetical EXCLUSIVE-OR Profile

We see that to accommodate the second quarter it is required to shift the $(n-2)$ -variable profile by $S = +1$ unit, and to accommodate the fourth quarter, the profile must be shifted by $S = -1$ unit. Thus, the absolute value of the difference in shift between the quarters in the first half and the quarters in the

second half, $\delta_S = 2$. Referring to figure 6, we see that for all two-variable consistent minimal profiles, the difference in shift $\delta_S = 0$. Referring now to the profiles in the first two columns of figure 4, the first profile represents the upper half and the second the lower half of the profile of the 3-variable function, 0 0 0 1 0 1 0 1. The shifts between the quarters for the profiles shown are zero for the first half and -1 for the second half, so that the difference in shifts $\delta_S = |0 - 1| = 1$, this function is consistent, and can be realized in a single input composite as shown in figure 5. It appears to be a general criterion that if the absolute value of the difference of shift, $\delta_S \geq 2$, the function is not consistent and requires more than one input composite for its realization. This criterion will be discussed further under partitioning.

THE PARTITIONING OF FUNCTIONS

Having examined some of the properties of the minimal, consistent two-variable solutions, we are now in a position to consider a method of partitioning the true terms of an arbitrary n-variable truth function into a set of functions each of which is consistent, in accordance with the design procedure laid out in the introduction. It was shown in the discussion of consistent functions that any term to term variations of the M vector within a span of four terms must be due only to variables x_1 and x_2 . Thus, if the function is divided into spans of four terms as in figure 2, the minimal two-variable solutions for each span considered independently must be compatible with all others if the function is consistent. Note that compatibility is not transitive, i. e. the fact that B is compatible with A and A with C does not imply that B is compatible with C.

It was conjectured, at first, that permuting every possible pair of variables with the variable x_1 and x_2 successively, leading to $n(n-1)/2$ related functions, (that is, related by having the same assumed physical embodiment, but altered by symmetry operations), and checking each of these derived functions for compatibility among its four bit spans would be a sufficient test for realizability. This conjecture turns out to be false for more than five variables. Since the condition is mathematically necessary, however, it can be used to delete incompatible bits from a non-realizable function. Although if the resulting partition function has more than five variables, it may still not be realizable. This process is called partitioning by the method of sieving, and a simple example is given in Table II.

In the light of the proposed criterion for realizability, it is possible to determine why the 2-variable sieving test breaks down. The criterion essentially states that if the input value of at least one of the variables, x_j , is not independent of the rows of the truth table but requires in one portion $q_{j1} N_{j1}$ and in another portion $q_{j2} N_{j2}$, and that if the absolute value of the difference between the input values, $\delta_S = |q_{j1} N_{j1} - q_{j2} N_{j2}| \geq 2$, the function is not realizable. The difference, δ_S , is determined by the shift required to accommodate a suitably

TABLE II
PARTITIONING OF A FIVE VARIABLE FUNCTION

Consistent Functions								First Partition					Perm.
State	F	A	B	C	D	E	F						
0	1	0	0	1	0	0	0	✓	✓	✓	✓	x_0	
1	0	0	0	0	0	0	0	1	1	1	0	x_4	
2	0	0	0	0				1	1	0	0	First Sieve	x_5
3	1	1	0	0				1	0	0	0		x_1
4	1	0	0	1				1	0	0	1	Solutions	x_2
5	1	1		0				I	B	C	A		x_3
6	0	0		0				B'	C'				
7	1	1		0									
8	0	0		0				The checked columns have compatible solutions. The true bits of the EXCLUSIVE-OR functions in columns 2 and 5 are treated as separate functions. The lower bit of column 5, representing a C' solution is stricken out.					Perm.
9	0	0		0									
10	0	0			0								
11	0	0			0								
12	1	0			1								
13	1	1			0			0	0	0	1	Second Sieve	x_0
14	1	0			1			0	0	0	0		x_3
15	1	1			0			1	0	0	1	Solutions	x_4
16	0	0			0	0		D'	0	B	0		x_1
17	1	0			0	1		1	0	1	0	Solutions	x_2
18	0	0				0		D'	0	B	0		x_5
19	0	0				0							
20	0	0					0						
21	1	1	0										
22	1	0	0				1						
23	1	1	0				0						
24	0	0	0					All columns compatible, all bits pass second sieve.					Perm.
25	1	0	1										
26	1	0	1					0	0	0	0	Solutions	x_0
27	1	0	1					0	0	1	0		x_1
28	0	0	0					0	0	0	0	Solutions	x_2
29	0	0	0					1	0	1	0		x_3
30	0	0	0	0	0	0	0	C'	0	D'	0	Solutions	x_4
31	1	1	0	0	0	0	0	C'	0	D'	0		x_5

Striking out the bits in the first row converts the solutions to those shown below the line. All of which are compatible.

TABLE II
(Cont'd)

PARTITIONING OF A FIVE VARIABLE FUNCTION

Taking the columns of the third sieve in order from the right, the first partition results in the function marked A, which is consistent. Its solution is:

$$\underline{S} = \begin{bmatrix} -3 \\ 3 \\ 1 \\ 2 \\ -1 \\ -1 \end{bmatrix}$$

The successive partitionings proceed in a similar manner.

chosen common profile to different portions of the function.

There are at least three ways in which $\delta_S \geq 2$ can occur:

$$(1) \quad q_{j1} = -1, \quad q_{j2} = +1$$

$$(2) \quad q_{j1} N_{j1} < q_k N_k < q_{j2} N_{j2} \quad \text{for} \quad q_{j1} = q_{j2} = q_k = \pm 1$$

$$(3) \quad q_k N_k \geq q_{j1} N_{j1} \geq q_{j2} (N_{j2} + 2) \geq q_l N_l \quad \text{or}$$

$$q_k N_k \leq q_{j1} N_{j1} \leq q_{j2} (N_{j2} - 2) \leq q_l N_l$$

for $q_k = q_{j1} = q_{j2} = q_l = \pm 1$ and such that no other input has a value intermediate between those of x_k and x_l .

Condition (1) is equivalent to complementing one of the inputs of a minimal compatible, 2-variable function. For example, complementing the variable X_1 in a C profile produces the B profile. This change is detected in at least one of the sieves.

Condition (2) in the sieves is equivalent to a permutation of the variables X_j and X_k . Thus, permuting the variables X_1 and X_2 in A-C profile gives rise to a C-D profile and since compatibility is not transitive, condition (2) is also detected in the sieving process.

Condition (3) cannot be detected in a sieve since only relationships such as $q_k N_k > q_j N_j$, $q_j N_j > q_l N_l$, $q_k N_k > q_l N_l$ would result, and would indicate an apparent compatibility.

The two variable sieving method is equivalent to the two monotonicity test for realizability^{77, 95}. Similar tests could be described which would test higher ordered monotonicities for functions of more than five variables but, since it is known that complete monotonicity is not sufficient for realizability and because the labor involved becomes great, it is not considered worthwhile.

A partitioning method can be based on the profile shift criterion mentioned above by taking successively higher ordered profiles as shown in figures 9a and 9b. This method partitioned Moore's function^{*} into two realizable functions, the one having only a single true state. The process also resulted in solutions for each of the functions.

CLASSIFICATION OF REALIZABLE FUNCTIONS

All of the four-variable single-input-composite switching circuits or solutions are tabulated in Table III along with the function in basic form which it represents.

* Moore's function is a completely monotonic function of 12 variables which is not realizable in a single threshold device.

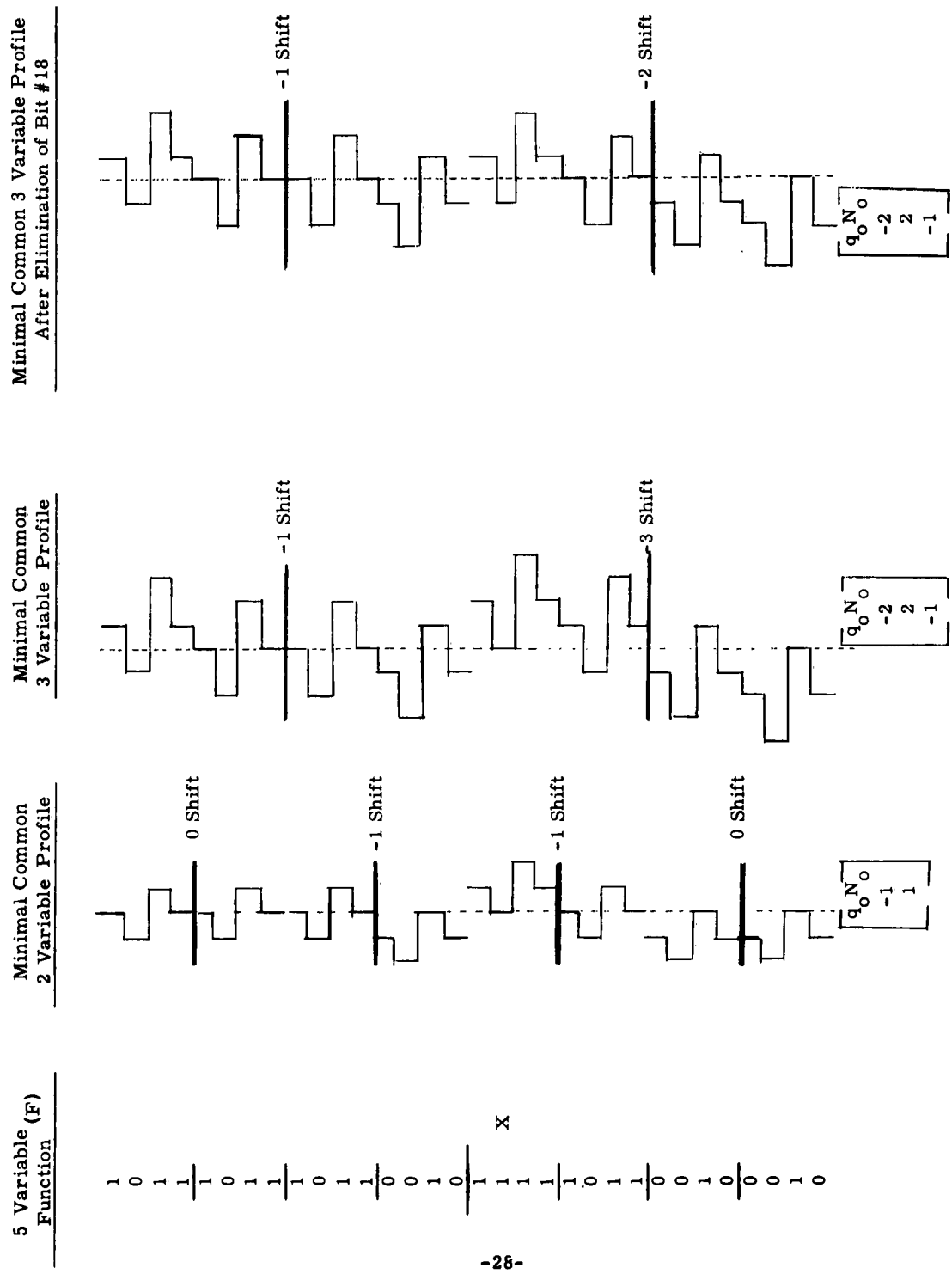


Figure 9a - Profile Shift Criterion for Realizability

The symbol $\binom{k}{m}$ stands for k true out of $m = 2^n$ states of the truth table. The solutions were extracted from a table of all four variable, combinational, linear-input switching circuits derived by R. C. Minnick by means of the simplex algorithm of linear programming using the Burroughs 220 computer.

Admitting the operation of inversion along with the symmetry operations, the fourteen functions tabulated suffice to map every realizable function. Inversion is equivalent to subtracting the given function from one which is true for every state. Whenever the number of true states of the realizable function is greater than 2^{n-1} , it is advantageous to invert. The solution is inverted by a vector subtraction, for example:

$$\begin{array}{ccccc}
 1 & - & F & = & F' \\
 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} & - & \begin{bmatrix} -1 \\ 2 \\ -3 \\ 1 \\ -2 \\ 1 \end{bmatrix} & = & \begin{bmatrix} 2 \\ -2 \\ 3 \\ -1 \\ 2 \\ -1 \end{bmatrix}
 \end{array} \tag{12}$$

Using the table for four variable functions, it is possible to build up a table for five variables by attaching one function to the end of another. A $\binom{6}{32}$ five variable basic function is formed by attaching a $\binom{1}{16}$ function to the end of a $\binom{5}{16}$ function. There are two choices of $\binom{5}{16}$ functions and we must take the one having a true state in its lower half. That is, the one on the right of the $\binom{5}{16}$ column of Table IV. Remembering that the input value of the new variable, X_5 , is negative and that $N_5 \geq N_4$ if the 5-variable function is to be in basic form, the first half of the lower 4-variable function must be contained in the second half of the upper function. Thus, since there are no true states in the lower half

	$\binom{1}{16}$	$\binom{2}{16}$	$\binom{3}{16}$	$\binom{4}{16}$	$\binom{5}{16}$	$\binom{6}{16}$	$\binom{7}{16}$	$\binom{8}{16}$
Solution	1	1	2	1 2	3 2	2 3	3 4	1 3 2
	-1	0	-1	0 -1	-1 -1	0 -1	-1 -1	0 -1 0
	-1	-1	-1	0 -1	-1 -1	-1 -1	-1 -2	0 -1 -1
	-1	-1	-2	-1 -1	-2 -1	-1 -2	-1 -2	0 -1 -1
	-1	-1	-2	-1 -2	-3 -1	-2 -2	-3 -3	-1 -2 -1
Function	1	1	1	1 1	1 1	1 1	1 1	1 1 1
	0	1	1	1 1	1 1	1 1	1 1	1 1 1
	0	0	1	1 1	1 1	1 1	1 1	1 1 1
	0	0	0	1 0	1 0	1 1	1 1	1 1 1
	0	0	0	0 1	1 1	1 1	1 1	1 1 1
	0	0	0	0 0	0 0	1 0	1 1	1 1 1
	0	0	0	0 0	0 0	0 0	1 0	1 1 0
	0	0	0	0 0	0 0	0 0	0 0	1 0 0
	0	0	0	0 0	0 1	0 1	0 1	0 1 1
	0	0	0	0 0	0 0	0 0	0 0	0 0 1
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0
	0	0	0	0 0	0 0	0 0	0 0	0 0 0

TABLE III - Primitive Solutions of Consistent,
Four-Variable Functions

of the $\binom{5}{16}$ function on the left hand, no true states can be added. Furthermore, since the same 4-variable profile must generate both halves of the 5-variable function, only compatible changes are allowed. These criteria are sufficient for all five variable basic realizable functions, but not for six variables, and in general, it is not known how to generate every realizable function for more than six variables. Table IV lists all realizable functions of 5-variables in basic form along with their $\underline{\Delta}$ and solution vectors. Note that all realizable functions of fewer than 5 variables are automatically included.

The $\underline{\Delta}$ vector indicates the number of essential variables in the function; if any of the terms $\Delta_i = k$ where k is the number of true states of the function, it means that the corresponding variables X_i in the basic solution are present only to suppress true states. Thus, they are given input weights, $q_i N_i = -N_0$.

A function of five or fewer essential variables can be tested for realizability by reference to the table. First, determine the $\underline{\Delta}$ vector for the given function. If the function has five or fewer essential variables, the table can be used.

Look for a matching set of Δ_i in the table, ignoring negative signs and the order of the terms, under the columns headed by $\binom{k}{32}$. If $k > 2^{n-1}$ look in the columns headed $\binom{k^*}{32}$, where $k^* = (2^n - k)$ for the basic form of the inverted function. The solution vector of the basic function (if the given function is realizable) is converted to the desired solution as in example 1. But, if the function was inverted, the basic solution is first inverted and then each input is complemented before deriving the \underline{D} and \underline{C} matrices.

[illegible]Table IV

Finally, the number of entries in Tables III and IV can be reduced by applying the concept of profile shifting. Suppose that the bias input value is not constant, but a quantity that can be varied positively and negatively in integer steps. The function that is generated will then depend upon the bias setting, and the number of different functions which can be generated depends upon the extent and complexity of the profile. It can be shown that by allowing the operation of profile shifting, only two special, non-minimal designs are required for all realizable, basic-form functions of three variables, three designs for four variables, and eight for five variables.

SECTION II

THE USE OF THE SIMPLEX ALGORITHM IN THE MECHANIZATION OF BOOLEAN SWITCHING FUNCTIONS BY MEANS OF MAGNETIC CORES

INTRODUCTION

In recent years many magnetic devices have been introduced which are used to mechanize switching functions. One class of devices which has received wide attention is the multi-aperture, or multi-path, structures, such as the Transfluxor,¹ the MAD's,⁹ and the Laddic.⁴² Their common property, in addition to using magnetic material with a rectangular hysteresis loop, is that some or all of the logical functions are performed by controlling the actual switching path through the structure. The philosophy is to achieve as much logic as possible within the structure, thereby reducing component count and interwiring.

The question arises, however, as to whether the simple toroidal core has been given sufficient consideration. A study of Karnaugh's paper, "Pulse Switching Circuits Using Magnetic Core,"⁵⁶ and an internal report by R. C. Minnick⁶⁵ revealed extremely interesting logical properties of the simple toroidal core based on the exploitation of its magnetic threshold. For example, the logical threshold function can be mechanized in a single core, including, of course, the AND and the OR functions which are special cases of the threshold function. This leads directly to the consideration as to how, if possible, does one synthesize any arbitrary Boolean function with a single core or perhaps several cores.

This section reports on a computer technique for mechanizing any arbitrary Boolean function with toroidal cores. In other words, the computer-derived solution will indicate the wiring configuration for a core or net of cores yielding a device for performing combinational logic. Switching functions of n variables are realizable in essentially one clock time.

The technique is a linear programming technique referred to as the Simplex algorithm. Of extreme importance is the fact that the logical designer may synthesize a function directly from the truth table without proceeding in the customary manner of expressing the function in Boolean canonical form and then attempting to minimize the hardware by algebraic manipulation or some other charting or mapping technique.

It will be interesting to note that this technique is not limited to magnetic cores but should apply to any device with a physical threshold.

NOTATION

The schematic representation of magnetic circuits is facilitated by the use of mirror symbols. In this paper cores (toroids) are represented by heavy vertical line segments, wire leads by horizontal line segments, and windings by 45° mirror symbols at the intersections of the cores and leads, figure 10. The sense of the magnetic field associated with a current in a given winding is obtained by "reflecting" the current in the winding mirror symbol. For example, if the core is initially in the reset, application of current I_1 to winding N_1 is in a direction to cause the core to set. If the core were initially set, the application of I_2 through N_2 would cause the core to reset. The magnetic logic in

connection with this paper is performed with a two-phase clock. The core is initially in the reset state. At phase one the core is set or not set in accordance with the logical requirements. At phase two time the core is reset. Signals induced by flux change are interpreted as logical ones.

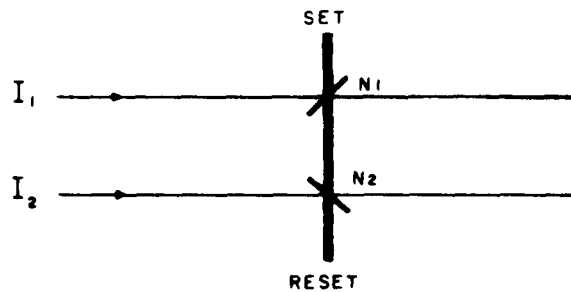


Figure 10 - Mirror Symbol Notation for Core Winding

THE VOTING FUNCTION

The voting function is a Boolean function which states that if μ or more out of n variables are present, then an output will result. For example, the truth table for $\mu = 2$ and $n = 3$ is shown in figure 11. This function may be mechanized with a single core. It is assumed that a unit current pulse is available to represent each variable plus its prime. Karnaugh's⁵⁶ rule for generating the voting function is to connect $(\mu - 1)$ of the primed variables in a direction to reset the core and $(n + 1 - \mu)$ unprimed variables in a direction to set the core. One way to wire the core is shown in figure 12. The m.m.f. equation is

$$F = I_2 N_2 + I_3 N_3 - I_1' N_1 \quad (13)$$

assuming that the magnetic threshold of the core is one ampere-turn and noting that all quantities in the m.m.f. equation are integers it may be stated that if

$$F_i \geq 1 \quad \text{then } f_i = 1$$

and $F_i \leq 0 \quad \text{then } f_i = 0$

where the index "i" denotes the row of truth table and F_i and f_i the corresponding values of the m.m.f. and Boolean function respectively. It is readily seen that if $N_1 = N_2 = N_3 = 1$ and if the values of the variables are applied to (13) in accordance with the truth table, the proper values of F_i are attained to satisfy the voting function required.

	I_3	I_2	I_1	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Figure 11 - Truth Table for Voting Function
where $\mu = 2$ and $n = 3$

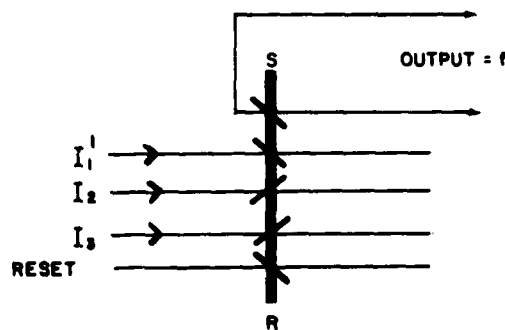


Figure 12 - Core and Wiring Configuration for Voting
Function where $\mu = 2$ and $n = 3$

It should be noted that a primed variable can be replaced by a unit current bias pulse, always present at phase one time, and by the unprimed variable in accordance with (14).

$$I_n' = I_o - I_n \quad (14)$$

Complementing is achieved since $I_n' = 1$ when $I_n = 0$ and when $I_n' = 0$, $I_n = 1$. Equation (13) may be written as

$$F = -I_o N_o + I_1 N_1 + I_2 N_2 + I_3 N_3$$

where N_o has the same number of turns as N_1 .

An extension to this type of switching function mechanization is to develop a technique for synthesizing any arbitrary Boolean function in a similar manner. It turns out that it is possible to mechanize many Boolean functions with a single toroid appropriately wound. Unfortunately, not all functions can be realized with a single core. For example, the "exclusive-OR" requires two cores. However, the method to be described will handle multicore solutions.

Consider the function 0 1 0 1 1 1 0 1 (1 denotes least significant bit). The truth table representation is shown in figure 13. If the core is wound according to (15), which means I_o and I_2

$$F = 2I_o + I_2 - 2I_1 - I_3 \quad (15)$$

applied to two turns and one turn respectively in a direction to set the core and I_1 and I_3 applied to two turns and one turn respectively in a direction to reset core, then the function is mechanized in a single core. This solution may be reached by intuition or by trial and error. When the number of variables

increases, the need for an algorithm becomes apparent.

THE SIMPLEX ALGORITHM

A linear programming technique referred to as Simplex provides an algorithm for determining the wiring configuration for mechanizing any arbitrary Boolean function with a single core where possible and with a multiplicity of cores when necessary. It is the aim of this paper to provide an insight to the workings of Simplex and its application to our problem, while at the same time avoiding the many associated details in connection with computation and programming. Consider the truth table of figure 13. An m.m.f. inequality may be written for each row. A threshold value of one ampere-turn is assumed for core switching, and in conditions where core switching is not desired the net m.m.f. is held less than or equal to zero ampere-turns.

	I_0	I_3	I_2	I_1	f
i=0	1	0	0	0	1
1	1	0	0	1	0
2	1	0	1	0	1
3	1	0	1	1	1
4	1	1	0	0	1
5	1	1	0	1	0
6	1	1	1	0	1
7	1	1	1	1	0

Figure 13 - Truth Table for $f = 01011101$

$$\begin{array}{rcl}
I_0 N_0 & \geq & 1 \\
I_0 N_0 + I_1 N_1 & \leq & 0 \\
I_0 N_0 + I_2 N_2 & \geq & 1 \\
I_0 N_0 + I_1 N_1 + I_2 N_2 & \geq & 1 \\
I_0 N_0 + I_3 N_3 & \geq & 1 \\
I_0 N_0 + I_1 N_1 + I_3 N_3 & \leq & 0 \\
I_0 N_0 + I_2 N_2 + I_3 N_3 & \geq & 1 \\
I_0 N_0 + I_1 N_1 + I_2 N_2 + I_3 N_3 & \leq & 0
\end{array} \tag{16}$$

The system of inequalities (16) is consistent (one or more solutions), and the function may be mechanized by a single core. If the system of inequalities (16) were inconsistent, then more than one core would be required.

If an arbitrary Boolean function is processed by Simplex, the results are as follows:

1. If the system of inequalities is consistent, a set of values for N_j will be found which satisfy the inequalities. However, it is important to note that a particular solution will be found which will minimize an arbitrarily specified linear function referred to as the objective or cost function of the form

$$Z = K_1 N_1 + K_2 N_2 + \dots + K_n N_n \tag{17}$$

where K are constants. (In other words the Simplex algorithm furnishes a method for solving a system of inequalities and at the same time provides a minimization criterion, namely the total number of turns.)

2. If the system of inequalities is inconsistent, then the Simplex solution

will indicate how the constraints must be modified and what portions of the desired function will be mechanized in each of the multiplicity of cores.

Before proceeding with the explanation of the Simplex algorithm, examples of solutions of two functions, a single core solution and two core solution, are given. The solution to the function 01011101 is $N_0 = 2$, $N_1 = -2$, $N_2 = 1$, and $N_3 = -1$. This of course results in (15). Consider the function, $h = 01100010$. The Simplex-derived solution is $N_1 = 1$, $N_2 = -1$, and $d_6 = K$. Noting the presence of $d_6 = K$ in the solution and temporarily ignoring it, the m.m.f. equation may be written,

$$F = I_1 - I_2 \quad (18)$$

A core wound in accordance with (18) yields the function $f = 00100010$. Note that f differs from h by its absence of a "one" in the 6th position. This corresponds to the subscript of d , a dummy variable, the role of which will be described later.

A new function, g , is formed by observing the dummy variables remaining in the solution and placing a "one" in the bit positions corresponding to the subscripts of the dummy variables. In this example $g = 01000000$. This function is now processed by Simplex, and the solution is $N_0 = -1$, $N_1 = -1$, $N_2 = 1$, $N_3 = 1$. The corresponding m.m.f. equation is

$$G = -I_0 - I_1 + I_2 + I_3 \quad (19)$$

Since no dummy variables remain in the solution to g , it is a one-core function.

In general g may not be a single-core function, and the procedure described above

would be repeated until the final function processed contained no dummy variables. In our example, two cores are required to mechanize h (refer to figure 14). Cores F and G are wound according to (18) and (19) respectively. An output wire is series connected to both cores. Either core F or G, or neither, will switch in accordance with the input lines energized. Hence, the voltage induced in the output line at phase two time represents the function h , or

$$h_i = f_i + g_i \quad (20).$$

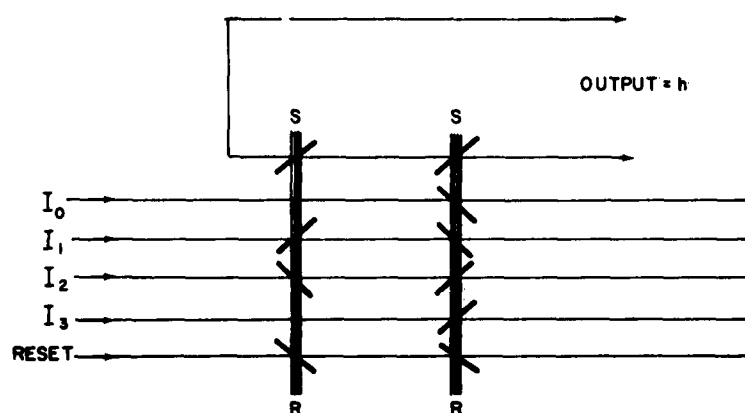


Figure 14 - Two-Core Mechanization
of $h = 011000010$

EXPLANATION OF SIMPLEX

Consider the following system of inequalities.

$$\begin{aligned} (1) \quad & a_{11}x_1 + a_{12}x_2 \leq b_1 \\ (2) \quad & a_{21}x_1 + a_{22}x_2 \leq b_2 \\ (3) \quad & a_{31}x_1 + a_{32}x_2 \geq b_3 \\ (4) \quad & a_{41}x_1 + a_{42}x_2 \leq b_4 \end{aligned} \tag{21}$$

If each inequality of (21) is treated as an equation and is graphed, the set of lines shown in figure 15 would result. Only the area included by the convex polygon abcd contains values of x_1 and x_2 which are solutions to (21). A useful theorem enables one readily to find a solution which will minimize an associated cost function.

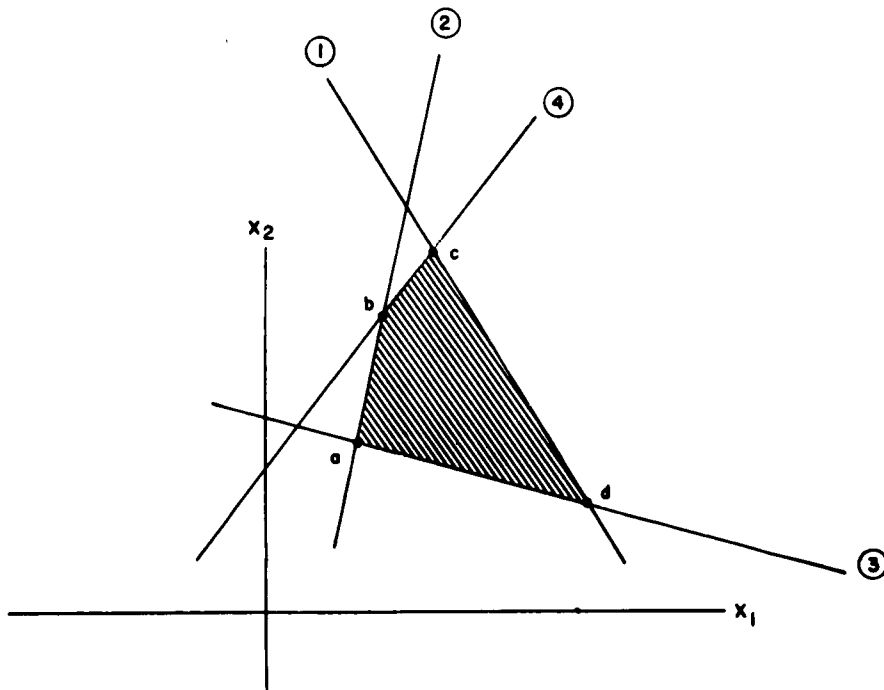


Figure 15 - Convex Polygon of Equations (21)

Theorem I:⁵⁷ A linear function defined over a convex polygon takes on its maximum and minimum value at a corner point of the convex polygon.

Hence, one simply tests the value of the cost function at each corner of the convex polygon to determine the values of x_1 and x_2 which satisfy the system of inequalities and minimizes the cost function. As the number of variables and equations increases, the geometrical picture becomes hopeless. It would amount to finding corner points of convex hulls in hyperspace. Fortunately, the Simplex algorithm need not be considered in the light of the above. The algorithm is perhaps best explained by example accompanied by pertinent theorems and definitions.

Consider the following system of inequalities and cost function.

$$\begin{aligned}
 (1) \quad & x_1 \geq 0 \\
 (2) \quad & x_2 \geq 0 \\
 (3) \quad & x_1 + 2x_2 \leq 8 \\
 (4) \quad & x_1 + x_2 \leq 5 \\
 (5) \quad & -x_1 + 2x_2 \leq 6
 \end{aligned} \tag{22}$$

$$Z = -2x_1 - 3x_2$$

The convex polygon is shown in figure 16. The value of the coordinate of the corner point (14, 15) minimizes the cost function.

The Simplex algorithm is utilized to find what is referred to as a basic feasible solution.

Definition I. A feasible solution is a solution which contains nonnegative x 's.

Definition II. A basic feasible solution is a solution which contains at most m positive x 's where m is the number of constraints.

Since the solutions are restricted to basic feasible solutions, the first two inequalities of (22) need not be written. The last three inequalities of (22) are converted to equations by the introduction of "slack" variables x_3 , x_4 and x_5 .

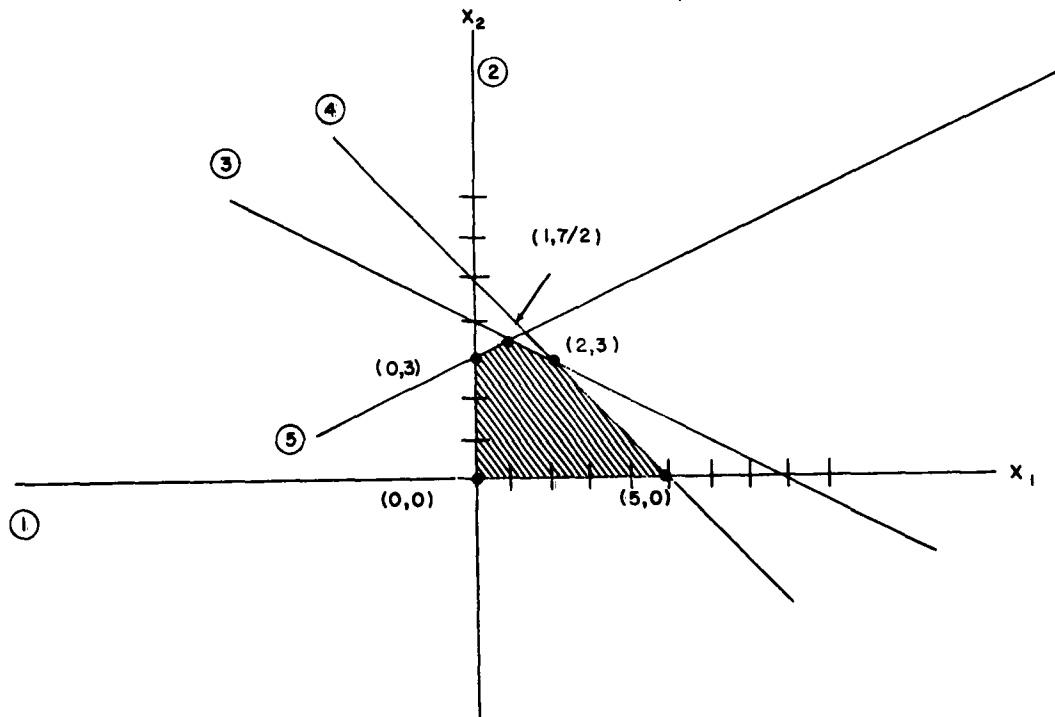


Figure 16 - Corner Points of the Convex Polygon of Equations (22)

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 8 \\ x_1 + x_2 + x_4 &= 5 \\ -x_1 + 2x_2 + x_5 &= 6 \end{aligned} \quad (23)$$

Equations (23) are now said to be in "canonical" form with respect to variables x_3 , x_4 , and x_5 . Notice that the coefficient of x_3 is plus one and that it does not appear in any other equation. This is true also for x_4 and x_5 . The canonical form in general is

$$\begin{aligned} x_1 + a_{1, m+1} x_{m+1} + \dots + a_{1n} x_n &= b_1 \\ x_2 + a_{2, m+1} x_{m+1} + \dots + a_{2n} x_n &= b_2 \\ \dots & \\ x_n + a_{n, m+1} x_{m+1} + \dots + a_{nn} x_n &= b_m \end{aligned} \quad (24)$$

Theorem II: If a problem has an optimum feasible solution, then it also has a basic feasible solution which is optimal.

Note that the canonical form of (23) enables one to obtain a basic, feasible solution simply by setting x_1 and x_2 equal to zero. One solution, then, is $x_3 = 8$, $x_4 = 5$, $x_5 = 6$, and the corresponding value of the cost function is $Z = 0$. The introduction of slack variables provides an initial basic feasible solution not necessarily optimum. By appropriate algebraic manipulation, (23) can be put into

canonical form with respect to any set of three x 's. These variables are referred to as "basic variables" or as the "basis." In this problem the total number of solutions to be examined is the combination of five things, taken three at a time, or ten solutions. For larger problems this trial and error procedure becomes prohibitive. Fortunately, there is a systematic iterative procedure for finding, in a relatively few number of steps, a solution which minimized the cost function.

Referring to the cost function of (23), if x_1 or x_2 were increased from zero, then the value of Z would be reduced. This, of course, is because their coefficients, called the "relative cost factors," are negative. First to be considered is x_2 , since it would appear that any change here would have the greater effect on the value of Z . (However, it is not essential that x_2 be given first consideration.) The question arises as to how large x_2 can be made without causing any of the other x 's to turn negative. Therefore, from (23) the following may be written:

$$\begin{array}{lll} 2x_2 = 8 & ; & \therefore x_2 = 4 \\ x_2 = 5 & ; & x_2 = 5 \\ 2x_2 = 6 & ; & \therefore x_2 = 3 \end{array}$$

The minimum positive value of x_2 is selected; i. e. $x_2 = 3$. If the value of 4 were selected, then x_5 would turn negative. If the value of 5 were selected, then both x_3 and x_5 would turn negative. The value of $x_2 = 3$ makes $x_5 = 0$. Hence, x_2 replaces x_5 in the basis and the equations (23) are now rearranged to canonical form with respect to the basic variables x_2, x_3, x_4 .

$$\begin{aligned}
2x_1 - x_5 + x_3 &= 2 \\
\frac{3x_1}{2} - \frac{x_5}{2} + x_4 &= 2 \\
-\frac{x_1}{2} + \frac{x_5}{2} + x_2 &= 3 \\
Z &= -\frac{7x_1}{2} + \frac{3x_5}{2} - 9
\end{aligned}
\tag{25}$$

The basic, feasible solution is $x_1 = 0$, $x_2 = 3$, $x_3 = 2$, $x_4 = 2$, and $x_5 = 0$ with the cost function decreasing to $Z = -9$. By examining the relative cost factors of (25), it can be seen that the value of the cost function can be decreased by increasing x_1 only. The positive value of the relative cost factor of x_5 prohibits increasing x_5 from zero since this would result in an increase for Z . Testing again to find the new basis, it can be seen that x_1 replaces x_3 .

$$\begin{aligned}
2x_1 &= 2 ; & \therefore x_1 &= 1 \\
\frac{3x_1}{2} &= 2 ; & \therefore x_1 &= \frac{4}{3} \\
-\frac{x_1}{2} &= 3 ; & \therefore x_1 &= -6
\end{aligned}$$

Notice that one value for x_1 is -6 . This cannot be used since it would result in an increase in the value of the cost function. Equations (25) are now put in canonical form with respect to the basic variables x_1 , x_2 , and x_4 .

$$\begin{aligned}
\frac{x_3}{2} - \frac{x_5}{2} + x_1 &= 1 \\
-\frac{3x_3}{4} + \frac{x_5}{4} + x_4 &= \frac{1}{2} \\
\frac{x_3}{4} + \frac{x_5}{4} + x_2 &= \frac{7}{2} \\
Z &= -\frac{x_5}{4} + \frac{7x_3}{4} - \frac{25}{2}
\end{aligned}
\tag{26}$$

The solution is $x_1 = 1$, $x_2 = \frac{7}{2}$, $x_3 = 0$, $x_4 = \frac{1}{2}$, and $x_5 = 0$; the value of Z is $-\frac{25}{2}$. Repeating the process

$$\begin{aligned}
-\frac{x_5}{2} &= 1 & ; & & x_5 &= -2 \\
\frac{x_5}{4} &= \frac{1}{2} & ; & & x_5 &= 2 \\
+\frac{x_5}{4} &= \frac{7}{2} & ; & & x_5 &= 14
\end{aligned}$$

The new basis is x_1 , x_2 , and x_5

$$\begin{aligned}
-x_3 + 2x_4 + x_1 &= 2 \\
3x_3 + 4x_4 + x_5 &= 2 \\
x_3 - x_4 + x_2 &= 3 \\
Z &= \frac{5}{2}x_3 + x_4 - 13
\end{aligned}
\tag{27}$$

The solution is $x_3 = 0$, $x_4 = 0$, $x_5 = 2$, $x_1 = 2$, and $x_2 = 3$. The value that Z takes is -13 . Note that now all of the relative cost factors are positive. Hence, neither x_3 nor x_4 can be increased. This indicates that the above solution is one which minimizes Z . This solution agrees with the solution obtained by the graphical method. Note that the intermediate solutions are also corners of the convex polygon of figure 16.

Since the Simplex solution yields only positive values for the unknowns, each N_j is represented as the difference of two positive numbers $(t_j - C_j)$ which would correspond to clockwise and counter-clockwise windings, respectively. Hence, the solution to function 01011101 would be of the form $t_0 = 2$, $C_1 = 2$, $t_2 = 1$, and $C_3 = 1$. Preparing inequalities (16) for Simplex they would be written as

$$\begin{array}{rclcl}
 (t_0 - C_0) & & & \leq & 1 \\
 (t_0 - C_0) & + & (t_1 - C_1) & \geq & 0 \\
 (t_0 - C_0) & & + & (t_2 - C_2) & \geq & 1 \\
 (t_0 - C_0) & + & (t_1 - C_1) & + & (t_2 - C_2) & \geq & 1 \\
 (t_0 - C_0) & & & + & (t_3 - C_3) & \leq & 1 & (28) \\
 (t_0 - C_0) & + & (t_1 - C_1) & & + & (t_3 - C_3) & \leq & 0 \\
 (t_0 - C_0) & & + & (t_2 - C_2) & + & (t_3 - C_3) & \geq & 1 \\
 (t_0 - C_0) & + & (t_1 - C_1) & + & (t_2 - C_2) & + & (t_3 - C_3) & \geq & 0
 \end{array}$$

where the t_j and C_j are the unknowns. The I_n coefficients are dropped because they have the value of unity.

Note that in addition to \leq signs appearing in (28) \geq also appear. In the case of \geq constraints a slack variable is subtracted and a dummy variable is added to the lefthand side of the inequality. Dummy variables are required to provide the initial basis. The initial basis is then comprised of slack and dummy variables, namely $S_0, d_1, d_2, d_3, S_4, S_5, d_6, d_7$.

$$\begin{aligned}
(t_0 - C_0) & + S_0 & = 1 \\
(t_0 - C_0) + (t_1 - C_1) & - S_1 + d_1 & = 0 \\
(t_0 - C_0) & + (t_2 - C_2) & - S_2 + d_2 = 1 \\
(t_0 - C_0) + (t_1 - C_1) + (t_2 - C_2) & - S_3 + d_3 & = 1 \\
(t_0 - C_0) & + (t_3 - C_3) + S_4 & = 1 \\
(t_0 - C_0) + (t_1 - C_1) & + (t_3 - C_3) + S_5 & = 0 \\
(t_0 - C_0) & + (t_2 - C_2) + (t_3 - C_3) - S_6 + d_6 & = 1 \\
(t_0 - C_0) + (t_1 - C_1) + (t_2 - C_2) + (t_3 - C_3) - S_7 + d_7 & = 0
\end{aligned} \tag{29}$$

The cost function is

$$Z = t_0 + t_1 + t_2 + t_3 + C_0 + C_1 + C_2 + C_3 + Md_1 + Md_2 + Md_3 + Md_6 + Md_7 \tag{30}$$

Note that dummy variables are included in the cost function. The coefficients M of the dummy variables are chosen very large to bias the problem so that, if possible, the dummy variables will not be included in the basis which minimizes Z^* .

* In some Simplex applications the existence of a dummy variable in the final basis indicates there is no feasible solution. In our application this type of result simply indicates that more than one core is required to mechanize the given function.

Equation (30) is rearranged to canonical form by appropriate substitutions from (29), eliminating the dummy variables.

$$\begin{aligned} Z = & (1 - 5M)t_0 + (1 - 3M)t_1 + (1 - 4M)t_2 + (1 - 2M)t_3 + (1 + 5M)C_0 \\ & + (1 + 3M)C_1 + (1 + 4M)C_2 + (1 + 2M)C_3 + MS_1 + MS_2 + MS_3 + MS_6 + MS_7 \\ & + 5M \end{aligned} \quad (31)$$

The problem is now ready to be treated in the manner previously described.

It will be recalled that if a function, h , is processed by Simplex, and dummy variables appeared in the solution, then a multi-core configuration was required to mechanize the function. Further, if one core were wound in accordance with the t_j and C_j resulting, part of the function was obtained and a new function was defined by placing a "one" in positions indicated by the subscripts of the dummy variables in the solution. Let us refer to the portion of the function initially mechanized as f , and the remaining portion as the residue function, g . Equation (29) may be written as follows:

$$\begin{aligned} \text{when } h_i = 1 \quad \text{then} \quad & \sum_j (t_j - C_j) - S_i + d_i = 1 \\ h_i = 0 \quad \text{then} \quad & \sum_j (t_j - C_j) + S_i = 0 \end{aligned}$$

Let $F_i = \sum_j (t_j - C_j)$; hence, when $F_i \geq 1$, $f_i = 1$, and $F_i \leq 0$, $f_i = 0$.

The symbol $\left[\begin{array}{c} \\ \end{array} \right]$ is defined as follows:

$$\begin{aligned} \left[X \right] &= 1 \quad \text{if} \quad X \geq 1 \\ \left[X \right] &= 0 \quad \text{if} \quad X \leq 0 \end{aligned}$$

The residue function is defined as follows:

$$[G_i] = 1 \quad \text{when } d_i > 0 \quad (32)$$

$$[G_i] = 0 \quad \text{when } d_i = 0 \text{ or when there is no } d_i \quad (33)$$

Equation (20) may now be written as

$$h_i = [F_i] + [G_i] \quad (34)$$

To prove (34) all possible combinations will be examined; i. e. :

$$\begin{aligned} h_i = 1 & \begin{cases} [F_i] = 1 \\ [F_i] = 0 \end{cases} \\ h_i = 0 & \begin{cases} [F_i] = 1 \\ [F_i] = 0 \end{cases} \end{aligned} \quad (35)$$

$$1. \ h_i = 1; \ [F_i] = 1 \quad \text{i.e.} \quad F_i \geq 1$$

Since Simplex guarantees that the original set of equations must be satisfied, we may write

$$F_i - S_i + d_i = 1$$

$$\text{But from (35)} \quad F_i \geq 1 \quad (36)$$

$$\text{also} \quad S_i \geq 0 \quad (\text{all variables are positive})$$

Adding (36)

$$2F_i + d_i \geq 2 \quad \text{or} \quad F_i \geq \frac{2}{2} - d_i \quad (37)$$

Hence where $h_i = 1$ and $F_i \geq 1$ the dummy variable must take the value of zero (all variables are positive) and we may say from (33) that $[G_i] = 0$.

$$\begin{aligned} 2. \quad h_i = 1; \quad [F_i] &= 0 \quad \text{i.e.} \quad F_i \leq 0 \\ F_i - S_i + d_i &= 1 \quad (38) \\ -F_i &\geq 0 \\ S_i &\geq 0 \end{aligned}$$

Adding (38)

$$d_i \geq 1 \quad \text{or} \quad d_i > 0$$

Hence, where $h_i = 1$ and $[F_i] = 0$ dummy variables must remain. From (32)

$$\begin{aligned} [G_i] &= 1 \\ 3. \quad h_i = 0; \quad [F_i] &= 1 \quad \text{i.e.} \quad F_i \geq 1 \\ F_i + S_i &= 0 \quad (39) \end{aligned}$$

In this case, (39) could be satisfied only if S_i were negative, which is not permitted by Simplex. Hence, the condition (15) is not possible.

$$4. \quad h_i = 0; \quad [F_i] = 0 \quad \text{i.e.} \quad F_i \leq 0$$

In this case (39) is satisfied by a positive slack variable which, of course, is permissible. However, since there is no d_i , we may say from (33), $[G_i] = 0$. Therefore, equation (34) is satisfied under all possible combinations.

The residue function defined above is now treated by Simplex. If it is a one-core function, the problem is terminated, yielding a two-core function. If, in processing the residue function, dummy variables remain in that solution, then a new residue function is defined.

The above procedure is repeated until the final residue function processed contains no dummy variables. Therefore, one magnetic core is required for each function or residue function processed in the course of a problem. From another point of view it may be said that the given function in a problem is reduced to a set of one-core functions, each of which might be represented by a consistent set of inequalities. To obtain the original function, all of these one-core functions are ORed together by the output line.

SIMPLEX MODIFICATIONS

In the course of the applications of the Simplex algorithm to the above problem, several difficulties arose which resulted in some modifications. For example, some solutions yielded minimum total number of turns at the expense of additional cores. This, under the requirements of unlimited total number of input turns, was not desirable. This type of answer, however, may be quite useful and will be discussed under future work. An extreme example of this type of difficulty is the result calling for zero turns and a residue function equal to the original function. The type of Simplex algorithm finally developed is "dummy-oriented;" i.e., in every applicable decision process during the course of the problem, dummy variables are removed from the basis. The following modifications were introduced:

1. Once a dummy leaves the basis it is not permitted to return. This is accomplished by ignoring the relative cost factors of dummy variables in the cost function.

2. The problem is not necessarily permitted to end when all of the relative cost factors become positive. Instead, the initial conditions of the problem are altered by changing the value of a relative cost factor assigned to one of the dummy variables so that, at this stage of the problem, one of the non-dummy variables assumes a negative relative cost factor. The test described previously is applied to determine which variable, if any, will leave the basis. It is possible that a variable with a negative relative cost factor cannot enter the basis because the test for minimum positive value yields only negative numbers. It should be noted that it is not difficult to change the initial conditions in the manner described above because only the cost equation is affected throughout the problem.

This procedure may be explained from the geometrical point of view. Consider the cost function of (22). If the cost function were changed to $Z = -2x_1 - 30x_2$, then a different corner of the convex polygon would minimize the new cost function.

Summarizing, it may be stated that the relative cost factors are repeatedly altered, forcing the problem to continue until a basis containing a minimum number dummy variables is obtained.

At the price of a more "expensive" solution, i. e., one which might contain more turns but uses less cores, the problem is permitted to terminate only when it is impossible to remove dummy variables from the basis.

It is interesting to note that it is always possible to remove at least one dummy from the initial basis. The partial result would be a function containing

a single "one" and a residue function with one less bit than the original function. All Boolean functions containing a single "one" are realizable with a single core. But once a particular dummy leaves the basis, all dummy variables in rows corresponding to ones which were consistent with the first dummy to leave the basis will also leave the basis.

VARIABLE SUPPRESSION

A method referred to as "variable suppression" has been developed. This method enables the synthesis of n-variable functions in terms of a known solution of functions of less than n variables. The variable-suppression technique yields solutions which may not be minimal, but this technique is one which is readily automatized.

This technique is best described by example. For simplicity, a function of four variables will be synthesized in terms of known solutions of functions of two variables. Consider the function $f = 1101001011001011$. The truth table is shown in figure 17. Note that the function f is divided into four subfunctions, a, b, c, and d. The portions of each function lined out by arrows are zeros. It will be observed that $f_i = a_i + b_i + c_i + d_i$. Hence, the synthesis of the subfunction will provide a solution for f simply by connecting the output winding of the cores associated with a, b, c, and d in series.

Consider function a. Bit positions $i = 0$ through $i = 3$ may be considered as a function of two variables, namely I_1 and I_2 . The solution for this function is

$$I_0 + I_1 - I_2 \tag{36}$$

	I_4	I_3	I_2	I_1	f	a	b	c	d
0	0	0	0	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	1
2	0	0	1	0	0	0	1	1	1
3	0	0	1	1	1	1	1	1	1
4	0	1	0	0	0	1	0	1	1
5	0	1	0	1	0	1	0	1	1
6	0	1	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	1	1
8	1	0	0	0	0	1	1	0	1
9	1	0	0	1	1	1	1	1	1
10	1	0	1	0	0	1	1	0	1
11	1	0	1	1	0	1	1	0	1
12	1	1	0	0	1	1	1	1	1
13	1	1	0	1	0	1	1	1	0
14	1	1	1	0	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1

Figure 17 - Truth Table for f and the Sub Functions

But it is not sufficient to wind core "a" with this input configuration because it represents a function of two variables which would repeat itself every four bit positions. It is necessary to suppress any ones that might appear in the remaining portion of the function.

Note that the most positive value (36) can take is 2. If a winding of two turns is provided for variable I_3 and negatively connected, then the presence of the I_3 current pulse, during bit positions $i = 4$ through $i = 7$, would cause zeros in these positions. Similarly, a winding for I_4 of two turns negatively connected would cause zeros in bit positions $i = 8$ through $i = 15$. Hence, the m.m.f. equation for "a" is

$$A = X_0 + X_1 - X_2 - 2X_3 - 2X_4$$

where

$$\begin{aligned} a_i &= 1 && \text{when } A_i \geq 1 \\ a_i &= 0 && \text{when } A_i \leq 0 \end{aligned}$$

The solution for bit positions $i = 4$ through $i = 7$ for function b is simply I_2 . Bit positions $i = 0$ through $i = 3$ are held to zero by one turn negatively connected for I_3' . The equivalent of I_3' is $I_0 - I_3$. Bit positions $i = 8$ through $i = 15$ are held to zero by one turn negatively connected for variable X_4 . The m.m.f. equation for b is

$$B = X_0 + X_2 + X_3 - X_4$$

Functions c and d are handled in the manner described above. Summarizing, the set of m.m.f. equations for mechanizing f are

$$A = X_0 + X_1 - X_2 - 2X_3 - 2X_4$$

$$B = -X_0 + X_2 + X_3 - X_4$$

$$C = -X_0 + X_1 - X_2 - X_3 + X_4$$

$$D = -3X_0 - X_1 + X_2 + 2X_3 + 2X_4$$

The cores and the winding arrangement are shown in figure 18.

The Simplex algorithm previously described was programmed and operated successfully on the Burroughs 220 Computer. This technique which will be referred to as Simplex Mod I provided the capability of handling switching functions of six variables. Computer storage capacity is the limiting factor with respect to the number of variables which may be processed.

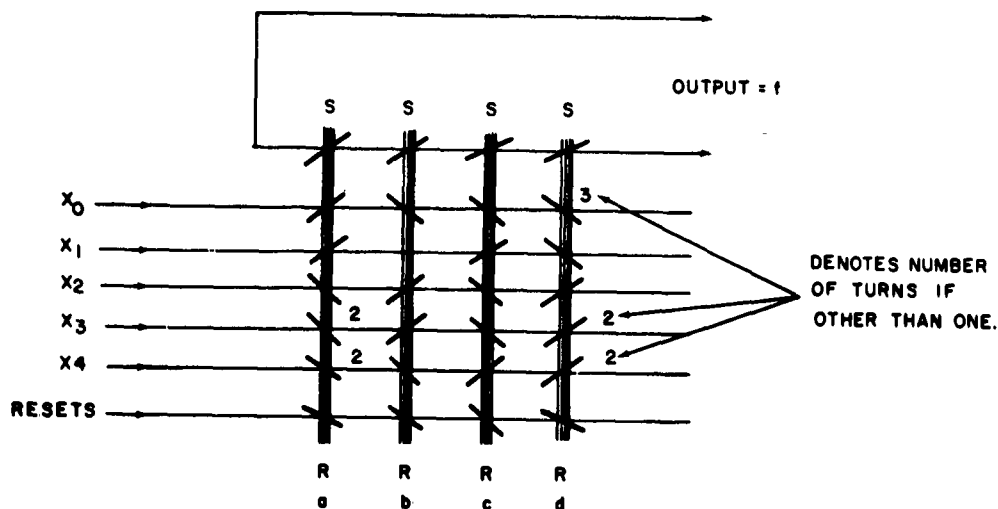


Figure 18 - Cores and Winding Arrangement for Mechanizing f

An improved version of Simplex (Simplex Mod II) has also been programmed for the 220 which, in addition to reducing computation time, has the capability of handling switching functions of nine variables. The modifications made to the former method effect the tableau structure, the computational procedure, and the cost function, and are described below.

TABLEAU STRUCTURE

It will be recalled that the constraining equations are the m.m.f. inequalities written for each row of the truth table. In order to provide for both clockwise and counterclockwise turns, each N_j is represented as the difference of two positives $(t_j - c_j)$ which correspond to clockwise and counterclockwise turns respectively. Consider the two-variable function $f = 1001$. The inequalities are:

$$\begin{aligned}(t_0 - c_0) & \geq 1 \\(t_0 - c_0) + (t_1 - c_1) & \leq 0 \\(t_0 - c_0) + (t_2 - c_2) & \leq 0 \\(t_0 - c_0) + (t_1 - c_1) + (t_2 - c_2) & \geq 1\end{aligned}\tag{37}$$

where t_j and c_j are the unknowns.

In forming an equation from an inequality containing the \leq sign, a slack variable is added. For those inequalities containing the \geq sign a slack variable is subtracted. At this point there is a deviation from Mod I in that a dummy variable is not also added to the left side of the equation. Setting up the constraints in this manner permits the S_i of the equations with \geq signs to take on negative values. Now the initial solution is a basic solution, rather than a basic feasible solution. Variables with negative values are permitted and given physical interpretation.

Considering this it follows that the initial basis is comprised of slack variables S_0, S_1, S_2, S_3 with constraining equations of the following form:

$$\begin{aligned}
(t_0 - c_0) & - S_0 = 1 \\
(t_0 - c_0) + (t_1 - c_1) & + S_1 = 0 \\
(t_0 - c_0) & + (t_2 - c_2) + S_2 = 0 \\
(t_0 - c_0) + (t_1 - c_1) + (t_2 - c_2) & - S_3 = 1
\end{aligned} \tag{38}$$

In the Mod I a cost function is specified. In Mod II a cost function is selected from one of the constraints which is of the form

$$S_i = \sum N_j - 1 \tag{38a}$$

Where the summation represents the total number of input turns (disregarding polarity) in the magnetic core and the minus one is the initial value of the slack variable, the quantity to be maximized. Only those slack variables with a value of minus one are eligible to be selected as cost functions. Constraints which are in contradiction will take on a value of ≤ -2 .

Consider the example $f = 1001$. The inequalities are

$$X_0 \geq 1 \tag{39}$$

$$X_0 + X_1 \leq 0 \tag{40}$$

$$X_0 + X_2 \leq 0 \tag{41}$$

$$X_0 + X_1 + X_2 \geq 1 \tag{42}$$

adding (39) and (42), (40) and (41), we get

$$\begin{aligned}
2X_0 + X_1 + X_2 & \geq 2 \\
2X_0 + X_1 + X_2 & \leq 0
\end{aligned} \quad \text{contradiction}$$

Adding the necessary slack variables to the inequalities we have:

$$X_0 - S_0 = 1 \quad (43)$$

$$X_0 + X_1 + S_1 = 0 \quad (44)$$

$$X_0 + X_2 + S_2 = 0 \quad (45)$$

$$X_0 + X_1 + X_2 - S_3 = 1 \quad (46)$$

Adding (43) and (46), (44) and (45), we get

$$2X_0 + X_1 + X_2 - S_0 - S_3 = 2 \quad (47)$$

$$2X_0 + X_1 + X_2 + S_1 + S_2 = 0 \quad (48)$$

subtracting 42 from 41

$$S_0 + S_1 + S_2 + S_3 = -2 \quad (49)$$

But S_2 and S_3 are initially positive quantities and are not permitted by the Simplex Method to take on negative values. Removing S_2 and S_3 from (49) we get

$$S_0 + S_3 \leq -2 \quad (50)$$

If either constraints (43) or (46) is optimized making S_0 or $S_3 \geq 0$, the burden of satisfying (50) is left to the other giving in each case $S_0 \leq -2$ or $S_3 \leq -2$, verifying that the form of the cost function must be as previously stated.

At first glance it might appear that in maximizing the value of S_i we are being forced to maximize ΣN_j since this is the only variable term in the cost function. It is obvious that this is undesirable, and it can be shown that not only can S_j be maximized without also maximizing ΣN_j , but that the minimum value of ΣN_j is achieved. Before this is shown, however, it might be best to explain why S_i

is being maximized and why the objective function takes the form of (38a).

It will be recalled that if a function, h , is processed by Simplex, and dummy variables appeared in the solution, then a multi-core configuration was required to mechanize the function. Further, if one core were wound in accordance with the t_j and c_j resulting, part of the function was obtained and a new function was defined by placing a "one" in positions indicated by the subscripts of the dummy variables in the solution. In Mod II dummies are replaced by negative slack variables. Refer to the portion of the function initially mechanized as f , and the remaining portion as the residue function, g . The constraints may be written as follows:

$$\text{when } h_i = 1 \text{ then } \sum_j (t_j - c_j) - S_1 = 1 \quad (51)$$

$$h_i = 1 \text{ then } \sum_j (t_j - c_j) + S_1 = 0 \quad (52)$$

Let $F_i = \sum_j (t_j - c_j)$: hence when $F_i \geq 1$, $f_i = 1$ and when $F_i \leq 0$, $f_i = 0$.

The symbol $\left[\begin{array}{c} \end{array} \right]$ is defined as follows:

$$\left[\begin{array}{c} X \end{array} \right] = 1 \quad \text{if } X \geq 1$$

$$\left[\begin{array}{c} X \end{array} \right] = 0 \quad \text{if } X \leq 0$$

The residue function is defined as follows:

$$\left[\begin{array}{c} G_i \end{array} \right] = 1 \quad \text{when } S_i < 0 \quad (53)$$

$$\left[\begin{array}{c} G_i \end{array} \right] = 0 \quad \text{when } S_i \geq 0 \text{ or when there is no } S_i \quad (54)$$

The function h is formed by

$$h_i = \left[\begin{array}{c} F_i \end{array} \right] + \left[\begin{array}{c} G_i \end{array} \right] \quad (55)$$

To prove (55) all possible combinations will be examined, i.e.,

$$\begin{array}{rcl}
 h_i = 1 & \begin{bmatrix} F_i \end{bmatrix} & = 1 \\
 & \begin{bmatrix} F_i \end{bmatrix} & = 0 \\
 \\
 h_i = 0 & \begin{bmatrix} F_i \end{bmatrix} & = 1 \\
 & \begin{bmatrix} F_i \end{bmatrix} & = 0 \\
 \\
 1. \quad h_i = 1; & \begin{bmatrix} F_i \end{bmatrix} & = 1 \quad \text{i.e., } F_i \geq 1 \quad (56)
 \end{array}$$

Since Simplex guarantees that the original set of equations must be satisfied, we may write

$$F_i - S_i = 1 \quad (57)$$

But from (56) $-F_i \leq -1$

$$\text{Adding (57)} \quad -S_i \leq 0 \quad (58)$$

Hence, where $h_i = 1$ and $F_i \geq 1$, the slack variable must be ≥ 0 and from (54), $[G_i] = 0$.

$$\begin{array}{rcl}
 2. \quad h_i = 1; & \begin{bmatrix} F_i \end{bmatrix} & = 0 \quad \text{i.e., } F_i \leq 0 \\
 & F_i - S_i & = 1 \quad (59) \\
 & -F_i & \geq 0
 \end{array}$$

adding (59) $-S_i \geq 1$ or $S_i < 0$

hence, where $h_i = 1$ and $[F_i] = 0$, negative slack variables must remain.

From (53), $[G_i] = 1$.

$$\begin{array}{rcl}
 3. \quad h_i = 0; & \begin{bmatrix} F_i \end{bmatrix} & = 1 \quad \text{i.e., } F_i \geq 1 \\
 & F_i + S_i & = 0 \quad (60)
 \end{array}$$

In this case, (60) could be satisfied only if S_i were negative which is not permitted* by Simplex. Hence, the condition (39) is not possible.

$$\begin{aligned}
 4. \quad h_i &= 0; \quad [F_i] = 0 \quad \text{i.e.,} \quad F_i \leq 0 \\
 F_i + S_i &= 0 \\
 -F_i &\geq 0
 \end{aligned} \tag{61}$$

adding (61)

$$S_i \geq 0.$$

This case is satisfied by a positive slack variable which of course is permissible. Since S_i is positive we may say from (54), $[G_i] = 0$. Therefore, equation (55) is satisfied under all possible combinations.

The residue function defined above is now treated by Simplex. If it is a one-core function, the problem is terminated, yielding a two-core function. If, in processing the residue function, negative slack variables remain in the solution, then a new residue function is defined.

The above procedure is repeated until the final residue function processed contains no negative slack variables. Therefore, one magnetic core is required for each function or residue function processed in the course of the problem. From another point of view, it may be said that the given function h is reduced to a set of one-core functions, each of which can be represented by a consistent set of inequalities. To obtain the original function, all of these one-core functions are ORed together by the output line.

From the above discussion it becomes obvious that it is desirable to satisfy

* Slack variables whose values are initially positive are not permitted to turn negative.

as much of the original function h as possible by successively maximizing as many cost functions (negative slack variables) as possible. Looking once again at the cost function

$$S_i = \sum N_j - 1$$

it becomes apparent that if in the maximizing process S_i is made zero, the corresponding h_i is satisfied. Specifically solving the first equation of (38) for S_1 , we get

$$S_1 = (t_0 - c_0) + (t_1 - c_1) - 1$$

which is the form of the objective function. It will later be shown that the maximum value that S_i can take is zero.

Setting up equations (38) for computational procedure, the following tableau is produced:

	h	N_0	N_1	N_2
S_0	-1	-1	0	0
S_1	0	1	1	0
S_2	0	1	0	1
S_3	-1	-1	-1	-1

For purposes of tableau interpretation it is convenient to imagine the equations re-written, so that all variables are on the left side and only the constants on the right hand side. The rows refer to the basic variables and the columns to the

non-basic variables. It is to be noticed that although care was taken in setting up the constraint equations to allow for positive and negative solutions by representing the N_j by the difference of two positive numbers $(t_j - c_j)$, this need not be carried into the tableau, but advantage can be taken of the fact that the coefficients of the t_j, c_j pairs in the non-basis are always numerically equal--but opposite in sign. Define $N_j = t_j$; $\bar{N}_j = c_j$. This definition plus the removal of the dummy variables of Mod I results in a dramatic reduction in the tableau size for Mod II. The tableau size for Mod I is given by Rows = 2^M and Col = $2(M+1) + K$. For Mod II given by Rows = 2^M and Col = $M + 1$, where M is number of variables and K is number of ones in the function. For example, a nine-variable problem containing 200 ones in the function requires 112,640 elements in the tableau for Mod I and 5,120 elements in the tableau for Mod II.

COMPUTATIONAL PROCEDURE

To simplify the explanation, and to arrive at a clear understanding of the modified Simplex process, a representation sample problem is examined to demonstrate the computational procedure. For reasons of computational ease and with no loss of generality a two-variable function with a multi-core solution will be used; specifically, let us consider $h = 1001$. The constraining equations are:

$$\begin{aligned}
 N_0^* - S_0 &= 1 \\
 N_0^* + N_1^* + S_1 &= 0 \\
 N_0^* + N_2^* + S_2 &= 0 \\
 N_0^* + N_1^* + N_2^* - S_3 &= 1
 \end{aligned}$$

where $N_j^* = (N_j - \bar{N}_j)$

From here on only one of the $N_j - \bar{N}_j$ pair will be carried.

By inspection it is evident that a basic solution is given by

$$N_0 = 0, N_1 = 0, N_2 = 0, S_0 = -1, S_1 = 0, S_2 = 0, S_3 = -1.$$

The constraints for the basic variables are expressed in terms of the non-basic variables and the initial tableau is set up as follows:

(a) $S_0 = N_0 \quad -1$

(b) $S_1 = -N_0 - N_1$

(c) $S_2 = -N_0 - N_2$

(d) $S_3 = N_0 + N_1 + N_2 -1$

(62)

cost
function

	h	N_0	N_1	N_2	
S_0	-1	-1	0	0	pivot row
S_1	0	1	1	0	
S_2	0	1	0	1	
S_3	-1	-1	-1	-1	

↑ pivot col.

Thus far, no objective function has been determined, and before proceeding any further one must be picked. Equation (62a) is chosen as the cost function because it is of the form previously described. It can be noted that equation (62d) is also of the proper form, but the first one encountered in scanning from top to bottom is always picked.

In order that a row be picked as the cost row in the tableau representation of the constraints it must contain:

- (1) A slack variable in the cost column of that row.
- (2) A minus one in the corresponding constant column element.
- (3) A negative element which has a non-slack variable row element.

At each step one of the non-basic variables is brought into the basis, for which the entry in the cost function has a negative sign. In this specific problem at this point N_0 is to be made basic. The column headed by N_0 will be the pivot column.

It should be noted here that it is possible and quite probable that there will exist more than one non-basic variable which would have these conditions. If row (d) is picked as the cost row, both N_0 and N_2 would be eligible. When this situation arises, the first variable encountered, going from left to right, is chosen.

That variable, (or one of those in case of a tie) which has the smallest positive ratio of the constant and the coefficient in the column thus chosen, must be removed from the basis; only slack variables are taken into consideration when determining the minimum ratio. This row is the pivot row. Again looking at the tableau S_1 will leave the basis since row two has a ratio of zero.

Going back to the constraining equations let us justify all that has been said thus far about the choice of variables for maximizing (62a). S_0 can only be maximized by increasing N_0 . If N_0 is increased, the values of S_1 , S_2 , and S_3 will change. Care must be taken at this point not to make either S_1 or S_2 negative. It must be understood that in this problem the slack variables which were added to the inequalities are sign-restricted. Therefore they must remain zero or assume positive values. Taking this into consideration, the largest value N_0 may assume is zero. In the attempt to maximize S_0 , no change is effected at this time. This is due to degeneracy which is inherent in our problem.

Since N_0 was selected to replace S_1 in the basis, the new basis will consist of S_0 , N_0 , S_2 , S_3 . To begin the next phase, these variables are solved in terms of the non-basic variables getting the new equations below. Using the rules for tableau transformation, the new tableau is derived.

$$\begin{aligned}
 (a) \quad S_0 &= -N_1 - S_1 - 1 \\
 (b) \quad N_0 &= -N_1 - S_1 \\
 (c) \quad S_2 &= +N_1 - N_2 + S_1 \\
 (d) \quad S_3 &= +N_2 - S_1 - 1
 \end{aligned} \tag{63}$$

	h	S_1	N_1	N_2
S_0	-1	1	1	0
N_0	0	1	1	0
S_2	0	-1	-1	1
S_3	-1	1	0	-1

Equation (63a) no longer appears in a form suitable to pick as a cost function. Correspondingly, the first row of the tableau fails to meet the requirements of a cost row because it does not contain a negative element with a non-slack corresponding variable row element. It must be remembered, however, that in order to reduce the size of the tableau only one part of N_j^* was chosen to appear and recalling also that $N_j = -\bar{N}_j$. Substituting $-\bar{N}_1$ for N_1 , the following is true

$$\begin{aligned}
 (a) \quad S_0 &= \bar{N}_1 - S_1 - 1 \\
 (b) \quad N_0 &= \bar{N}_1 - S_1 \\
 (c) \quad S_2 &= -\bar{N}_1 - N_2 + S_1 \\
 (d) \quad S_3 &= N_2 - S_1 - 1
 \end{aligned} \tag{64}$$

Cost Function	h	S_1	\bar{N}_1	N_2
S_0	-1	1	-1	0
N_0	0	1	-1	0
S_2	0	-1	+1	1
S_3	-1	1	0	-1

\leftarrow 2 Pivot Row
 \uparrow 2 Pivot Column

Equation (64a) can again be selected as our cost function.

This time \bar{N}_1 will be brought into the basis and S_2 removed. Again there can be no increase in the cost function because the minimum ratio is zero.

Once more expressing the basic variables in terms of the non-basic variables and transforming the tableau, we have:

$$\begin{array}{ll}
 \text{(a) } S_o = -N_2 & -S_2 \quad -1 \\
 \text{(b) } N_o = -N_2 & -S_2 \\
 \text{(c) } \bar{N}_1 = -N_2 + S_1 & -S_2 \\
 \text{(d) } S_3 = N_2 - S_1 & -1
 \end{array} \quad (65)$$

	h	S_1	S_2	N_2
S_o	-1	0	1	1
N_o	0	0	1	1
\bar{N}_1	0	-1	1	1
S_3	-1	1	0	-1

The first row of the tableau has again lost its identity as a cost row. This is so because N_2 has a negative coefficient or cost factor. As before, use is made of the relationship $N_2 = -\bar{N}_2$ and the necessary changes give:

$$\begin{array}{ll}
 \text{(a) } S_o = \bar{N}_2 & -S_2 \quad -1 \\
 \text{(b) } N_o = \bar{N}_2 & -S_2 \\
 \text{(c) } \bar{N}_1 = \bar{N}_2 + S_1 & -S_2 \\
 \text{(d) } S_3 = -\bar{N}_2 - S_1 & -1
 \end{array} \quad (66)$$

	h	S_1	S_2	\bar{N}_2	
S_o	-1	0	1	-1	cost function
N_o	0	0	1	-1	pivot row
N_1	0	-1	1	-1	
S_3	-1	1	0	1	

This time row one will be both cost row and pivot row since the minimum positive ratio occurs in this row. In this transformation, S_o leaves the basis and therefore assumes the value of zero.

Previously, it was stated that in maximizing S_i the ΣN_j^* would assume the minimum value possible. It will be shown that the cost slack S_i can never assume a value greater than zero. Therefore, ΣN_j^* cannot increase by more than one since S_i must originally be equal to minus one.

First take the situation where the cost row becomes the pivot row. In this case, the cost slack will leave the basis and have a value of zero.

In the only other situation, the cost row is not also the pivot row. In this case if the cost is to change, then the constant column element of the pivot row must be non-zero. From (67) it is seen that in order for V_i to be picked as the pivot row

$$\frac{K_i}{E_i} < \frac{K_c}{E_c} \quad \text{where } K_i, E_i, K_c, E_c \text{ are positive numbers.}$$

From the transformation rules, the New Cost (K_c) is received.

Where

$$\begin{aligned} \text{new } K_c &= -\text{old } K_c - \frac{(-E_c \cdot K_i)}{E_i} \\ &= -\text{old } K_c + \frac{E_c \cdot K_i}{E_i} \end{aligned}$$

$$\text{but old } K_c < \frac{E_c \cdot K_i}{E_i} \quad \text{keeping the cost negative.}$$

Concluding, it is evident that the cost must remain negative unless the cost row is picked as the pivot row, in which case the cost goes to zero.

S_c	$-K_c$	$-E_c$	cost row
V_i	K_i	E_i	pivot row

(67)

Vari.
Col.

Con.
Col.

Returning to the computation and taking the next step--once again the equations are put into canonical form with respect to \bar{N}_2 which was chosen to enter the basis, and the tableau is transformed giving:

- (a) $\bar{N}_2 = S_o + S_2 + 1$
- (b) $N_o = S_o + 1$
- (c) $\bar{N}_1 = S_o + S_1 + 1$
- (d) $S_3 = -S_o - S_1 - S_2 - 2$

	h	S_1	S_2	S_o
\bar{N}_2	1	0	-1	-1
N_o	1	0	0	-1
\bar{N}_1	1	-1	0	-1
S_3	-2	1	1	1

Looking at the equations and tableau, it is obvious that a new cost function can't be picked. Observing also that S_3 with a value -2 remains, which indicates a multi-core solution with the first core satisfying only the first three states of the truth function, leaving a residue function $g_1 = 0001$.

The solution for the first core:

$$\begin{aligned}N_o &= 1 \\ \overline{N}_1 &= 1 \\ \overline{N}_2 &= 1\end{aligned}\tag{69}$$

DON'T-CARE CONDITION

The "don't care" condition is readily handled by the Simplex algorithm. This is accomplished by removing the constraints which involve input conditions which never occur or are of no consequence. In many cases tested, this results in a marked reduction in hardware. Further, since a given function is mechanized by OR-ing together one-core functions, a "don't care" list may be generated for any problem. This is accomplished by using the positions of the "ones" in the sub-function, K , as the "don't care" conditions in sub-function, $K + 1$, and the positions of the ones in sub-functions K and $K + 1$ and "don't cares" for sub-function $K + 2$, etc. This procedure also in many cases resulted in a reduction of hardware. For example, a nine-variable function processed in the design of the decimal adder was reduced from a 24-core to a 16-core design.

SECTION III

CONCLUSIONS

The study program, magnetic logical transducers, is concerned with the problem of achieving designs for single stage, combinatorial embodiments of arbitrary switching functions of n variables by means of magnetic phenomena. Because of their inherent simplicity and reliability, and because the types of functions which they can embody do not depend upon physical structure as in the case of multipath magnetic devices, the study is confined to square-loop magnetic toroids; the required versatility in function embodiment is attained by applying the, at first, rather nebulous concept of threshold device logic.

The study necessarily divided itself into two complementary phases, the first concerned with an immediate solution to the design problem and the second with a theoretical study of threshold devices having linear summing input circuits. The result of the first phase is an algorithm in the form of a computer program written for the Burroughs 220 computer. Functions of as many as nine variables are routinely designed in a form which is approximately minimal, at least in core count, by the use of this algorithm. The result of the second phase is a greater understanding of the operation of the input circuit of a threshold device, logic element, and a new tool, the function profile, which makes possible the design of functions of any number of variables, limited only by the amount of labor involved. Furthermore, an approach to the formulation of a logic of functions realizable in a single threshold element is made possible

by the discovery of minimum sets of circuits which map all realizable functions of a given number of variables and of automatic means for describing all required function transformations. The concept of function modification by profile shifting appears to have application to the field of Bionics.

SECTION IV
RECOMMENDATIONS

It is recommended that study continue in the areas described below.

THEORETICAL STUDIES

The Profile Technique is a powerful tool which is used to study the input composite of a magnetic core or other threshold devices. The areas of interest include (a) the formulation of a realizability test algorithm which is necessary and sufficient for any number of variables, (b) proof of all theorems used in the partitioning algorithm, (c) the development of an algorithm for reducing an arbitrary solution to minimality, (d) study of sets of building blocks based on profile shifting.

SIMPLEX

Minimality

Though satisfactory designs are achieved by Simplex, they are not, in general, minimal, where minimal is defined as the smallest number of threshold elements required to mechanize a given function. The difficulty is that the manner in which a switching function may be partitioned into threshold functions is not unique. It has been demonstrated that the threshold functions selected to form the specified switching function in many cases depends on the starting point chosen in a given problem. It has also been demonstrated that a reduced threshold element count can result by introducing an increasing "don't care" list to each problem, i.e. the bit positions of partition k become the "don't care"

conditions for partition $k+1$. This is permissible because the partitioned functions are inclusively ORed. This procedure is equivalent to mapping the required function by sets which are permitted to overlap. However, to produce minimal solutions requires the development of criteria which lead to a minimal number of sets to map the function. Preliminary thinking indicates that the realizability tests based on the Hamming distance between bits may provide a good starting point for such an analysis. Study is proposed in this area.

Input Limitations

From an engineering point of view, based on tolerance requirements, * an upper limit may be placed upon the total input weights and the threshold value. The nature of the Simplex algorithm is such that these additional constraints may be introduced into the existing program. This is possible since in forming a partition the input solution is available for each intermediate step. A solution may be chosen which contains the maximum number of bits and which is still within the given threshold value and total input weight specification. Though the overall element count may be increased, feasible circuits are designed. It is proposed that the above feature be considered for future study.

Experimental Studies

Any foreseeable system using magnetic threshold logic is likely to be composed of multi-level nets. For this reason, it is necessary to study the terminal properties of a magnetic core with various kinds of loads to develop systems with fan-in and fan-out capabilities. Newer magnetic materials with smaller H_c and/or sharper thresholds may provide break-through.

*Chao, J. C., On Driver Tolerances for Linear Input Logic with Magnetic Cores, Burroughs Corp. internal publication, RM 61-25 (in preparation).

BIBLIOGRAPHY

(By Author)

1. Abbott and Suran, "Multihole Ferrite Core Configuration and Application", PROCEEDINGS OF THE I.R.E., VOL. 45, August 1957, pp 1081-93.
2. Abhyankar, S., "Absolute Minimal Expressions of Boolean Functions", I.R.E. TRANSACTIONS ON ELECTRONIC COMPUTERS, VOL. EC-8, No. 1, March 1959, pp 3-8.
3. Adams, C. J. and Cooper, R. E., "Pulse-Type Logic-Element Industrial Control", NATIONAL ELECTRONICS CONFERENCE PROCEEDINGS, VOL. 13, 1957, pp 898-913.
4. Andrews, L. J., "A Technique for Using Memory Cores as Logical Elements", EASTERN JOINT COMPUTER CONFERENCE, December 10-12, 1956, PROCEEDINGS, N.Y., A.I.E.E., 1957, pp 39-46.
5. Anonymous, "Magnetic Amplifiers for Digital Computers", INSTRUMENT PRACTICE, VOL. 12, May 1958, pp 518-520.
6. Auricoste, J., Lhomme, H., "Les Circuits Logiques Magnetiques: Application aux Commandes a Sequences Industrielles", ONDE ELECTRIQUE, VOL. 38, n 372, March 1958, pp 217-24.
7. Baldwin, J. A., Jr. and Rogers, J. L., "Inhibited Flux - A New Mode of Operation of the Three-hole Memory Core", JOURNAL OF APPLIED PHYSICS, Supplement to VOL. 30, April 1959, pp 58S-59S.
8. Bamrough, B., "A Digital Computer Based on Magnetic Circuits", I.E.E. CONVENTION ON FERRITES, London, 1956, (supplements 5-7 to Part B. of I.E.E. PROCEEDINGS, VOL. 104, 1959), pp 485-490.
9. Bennion, David, Crane, Hewitt D., and Heinzmann, Fred., "Multi-aperture Magnetic Devices for Computer Systems", Stanford Research Institute, TECHNICAL REPORT 2, October 1957, p 130.

10. Bennion, David, Crane, Hewitt D., and Heinzmann, Fred., "Multi-aperture Magnetic Devices for Computer Systems", FINAL REPORT, January 1958, p 24.
11. Birkhoff and MacLane, A SURVEY OF MODERN ALGEBRA, New York, MacMillan, 1946, p 227.
12. Bloch, E. and Paulsen, R. C., "Magnetic Core Logic in a High-Speed Card-to-Tape Converter", I.R.E. TRANSACTIONS ON ELECTRONIC COMPUTERS, VOL. EC-8, No. 2, June 1959, pp 169-181.
13. Bobeck, Andrew H., "A New Storage Element Suitable for Large-Sized Memory Arrays - The Twistor", THE BELL SYSTEM TECHNICAL JOURNAL, VOL. 36, November 1957, pp 1319-1340.
14. Bobeck, Andrew H. and Fischer, Robert F., "Reversible, Diodeless, Twistor Shift Register", JOURNAL OF APPLIED PHYSICS, Supplement to VOL. 30, April 1959, pp 43S-44S.
15. Bonn, T. H., "Analysis of Magnetic Switching Circuits", PROCEEDINGS OF THE INTERNATIONAL SYMPOSIUM ON THEORY OF SWITCHING, April 1957.
16. Bonn, T. H., "The Ferractor", CONFERENCE ON MAGNETISM AND MAGNETIC MATERIALS, October 16-18, 1956, Boston, Mass., pp 654-665.
17. Briggs, G. R., "A Magnetic-core Gate and its Application in a Stepping Register", Massachusetts Institute of Technology, Digital Computer Laboratory, Cambridge, October 30, 1952, p 14 (Engineering Note E-475).
18. Brown, D. R., "Magnetic Material for High Speed Pulse Circuits", A.I.E.E., WINTER GENERAL MEETING, New York, N. Y., January 1953.
19. Buck, Dudley A., "Magnetic and Dielectric Amplifiers", Massachusetts Institute of Technology, Digital Computer Laboratory, Cambridge, August 28, 1952, p 13, (Engineering Note E-477).
20. Buck, Dudley A., "Binary Counting with Magnetic Cores", Massachusetts Institute of Technology, Digital Computer Laboratory, Cambridge, December 6, 1951, p 18, (Engineering Note E-438).

21. Burroughs Corporation, Research Center, "Digital Magnetic Circuits; Theory, Electrical Design and Logical Design", VOL. 1, 2, and 3, April 1959.
22. Chen, Mao-Chao, "A Magnetic Core Parallel Adder", I.R.E. TRANSACTIONS ON ELECTRONIC COMPUTERS, VOL. EC-7, December 1958, pp 262-264.
23. Chow, C. K., "Boolean Function Realizable with Single Threshold Devices", PROCEEDINGS OF THE I.R.E., VOL. 49, No. 1, January 1961, pp 370.
24. Cornell, W. A., "A Special Purpose Solid State Computer Using Sequential Access Storage", PROCEEDINGS OF THE WESTERN JOINT COMPUTER CONFERENCE, May 6, 7, 8, 1959, Los Angeles.
25. Crane, H. D., "A High-Speed Logic System Using Magnetic Elements and Connecting Wire Only", I.R.E. PROCEEDINGS, VOL. 47, January 1959, pp 63-73.
26. Crane, H. D., "A High-Speed Logic System Using Magnetic Elements and Connecting Wire Only", PROCEEDINGS, NONLINEAR MAGNETICS AND MAGNETIC AMPLIFIERS, SPECIAL TECHNICAL CONFERENCE, August 1958, pp 465-482.
27. Crane, H. D., "Multi-Aperture Magnetic Devices for Computer Systems", SPECIAL REPORT, Stanford Research Institute, February 1957, p 44.
28. Dorfman, R; Samuelson, P. A.; Solow, R. M., LINEAR PROGRAMMING AND ECONOMIC ANALYSIS, McGraw-Hill Book Company, Inc., New York, Toronto, London, 1958, p 78.
29. Dumaire, Marc; Jeudon, Herveline; Lilimand, Monique, "Logical Circuits Using Magnetic Cores and Capacitors (Diodeless)", PROCEEDINGS SPECIAL TECHNICAL CONFERENCE, NONLINEAR MAGNETICS AND MAGNETIC AMPLIFIERS, August 1958, p 243.
30. Dunnet, W. J. And Lemack, A. G., "Transistor-Magnetic Core Biological Computer Element", WESTERN JOINT COMPUTER CONFERENCE, March 1959.

31. Eldridge, D., "A New High-Speed Digital Technique for Computer Use", PROCEEDINGS, INSTITUTION OF ELECTRICAL ENGINEERS, Paper 2672M, published June 1958, p 8. To be republished in VOL. 106B (1959).
32. Ellis, David and Andrews, Lad, "Inverted-Core Logic and the Synthesis of Boolean Functions Therefrom", RESEARCH REPORT, NCR-ED BULLETIN NO. 111, National Cash Register Company, Hawthorne, California, October 1956. EASTERN JOINT COMPUTER CONFERENCE, December 10-12, 1956, p 46.
33. Englebart, D. C., "A New All-Magnetic Logic System Using Simple Cores", DIGEST OF TECHNICAL PAPERS, 1959, Solid State Circuits Conference, February 12-13, 1959, p 66.
34. Evans, W. G., Hall, W. G., and VanNice, R. I., "Magnetic Logic Circuits for Industrial Control Systems", AMERICAN INSTITUTE OF ELECTRICAL ENGINEERS, TRANSACTIONS Part II, VOL. 75, 1956, pp 166-71.
35. Frauck, A; Marette, G. F.; and Parsegyan, B. I., "Deposited Magnetic Films as Logic Elements", PROCEEDINGS OF THE EASTERN JOINT COMPUTER CONFERENCE, December 1-3, 1959, Boston, Massachusetts, pp 28.
36. Freedman, A. L., "Magnetic Core Matrices for Logical Functions", ELECTRONIC ENGINEERING, VOL. 31, June 1959, pp 358-361.
37. Frei, E. H., and Treves, D. (The Weizmann Institute of Schienc, Rehovet, Israel), "The Crossed Fields Magnetic Transducer (Amplifier) and its Application", CONFERENCE ON MAGNETISM AND MAGNETIC MATERIALS, October 16-18, 1956, Boston, Massachusetts, pp 666-672.
38. Ganzhorn, K., "Fundamental Magnetic Logical Networks in Computers", ELEKTRONISCHE RUNDSCHAU, VOL. 11, August 1957, pp 229-34 (In German).
39. Gianola, U. F., "Integrated Magnetic Circuits for Synchronous Sequential Logic Machines", BELL SYSTEM TECHNICAL JOURNAL, VOL. 39, No. 2, March 1960, pp 295-332.
40. Gianola, U. F., "Nondestructive Memory Employing a Domain Oriented Steel Wire", JOURNAL OF APPLIED PHYSICS, VOL. 29, May 1958, pp 849-853.

41. Gianola, U. F., "Possibilities of all Magnetic Logic", JOURNAL OF APPLIED PHYSICS, Supplement to VOL. 32, No. 3, March 1961, pp 275.
42. Gianola, U. F., "The Laddic - A Magnetic Device for Performing Logic", BELL SYSTEM TECHNICAL JOURNAL, VOL. 38, January 1959, pp 45-72.
43. Gianola, U. F. and Crowley, T. H., "Magnetic Networks for Performing Logic", PROCEEDINGS, TECHNICAL CONFERENCE ON NONLINEAR MAGNETICS AND MAGNETIC AMPLIFIERS, Los Angeles, California, August 6-8, 1958.
44. Gianola, U. F. and James, D. B., "Ferromagnetic Coupling Between Crossed Coils", JOURNAL OF APPLIED PHYSICS, VOL. 27, June 1956, pp 608-609.
45. Gillert, H., "Magnetic Switching Circuits for Representing Logical Combinations", NACHRICHTENTECH Z. (N.T.Z.) VOL. 10, No. 8, August, 1957, pp 391-402, (In German).
46. Gray, H. J. Jr., Lipkin, D. M. and Thibodeau, L. E., "Investigation of Transistor and Trnasistor/Magnetic Core Circuitry for High Speed Digital Computer Application", Remington Rand UNIVAC Div., Sperry Rand Corp., April 30, 1956, p 62. (Report No. AFCRC TN-56-184) (Contract AF 19(604)1376), Unclassified report.
47. Guenther, W. E., Noreen, D. H. and McNeill, J. H., "Design Considerations of a Static Logic Annunciator", NATIONAL ELECTRONICS CONFERENCE PROCEEDINGS, VOL. 13, 1957, pp 914-925.
48. Guterman, S., Kodis, R. D. and Ruhman, S., "Circuits to Perform Logical and Control Functions with Magnetic Cores", I.R.E. CONVENTION RECORD, 1954, Part 4, pp 124-132.
49. Guterman, S. Kodis, R. D. and Ruhman, S., "Logical and Control Functions Performed with Magnetic Cores", I.R.E. PROCEEDINGS, VOL. 43, March 1955, pp 291-8.
50. Hamming, R. W., "Error Detecting and Error Correcting Codes", BELL SYSTEM TECHNICAL JOURNAL, VOL. 26, April 1950, pp 147-160.

51. Harvard Computation Laboratory, "Synthesis of Electronic Computing and Control Circuits", ANNALS OF THE HARVARD COMPUTATION LABORATORY, VOL. 27, Harvard University Press, 1951.
52. Haynes, M. K., "Model of Nonlinear Flux Reversal of Square-loop Polycrystalline Magnetic Cores", JOURNAL OF APPLIED PHYSICS, VOL. 29, 1958 (February or March), p 472.
53. Higonnet and Grea, LOGICAL DESIGN OF ELECTRICAL CIRCUITS, McGraw-Hill Book Company, Inc., 1958, p 80.
54. Hunter, L. P. and Bauer, E. W., "High Speed Coincident Flux Magnetic Storage Principles", JOURNAL OF APPLIED PHYSICS, VOL. 27, No. 11, November 1956, pp 1257-1261.
55. International Business Machines Corp., Kingston, N. Y., "Magnetic Core Logic Improvement Program", FINAL REPORT, January 1, 1958, IV, (Contract AF 30(635)3130), Unclassified report, TECHNICAL ABSTRACT BULLETIN, May 15, 1958, p 608.
56. Karnaugh, M., "Pulse Switching Circuits Using Magnetic Cores", PROCEEDINGS OF THE I.R.E., May 1955, VOL. 43, pp 510-584.
57. Kemeny, J. G., Snell, J. L., Thompson, G. L., INTRODUCTION TO FINITE MATHEMATICS, Prentice-Hall, Inc., Englewood Cliffs, N. H., pp 256-257.
58. Koblents, I. G., "Design and Analysis of Single-Crystal Magnetic Logic and Trigger-Commutation Networks", ELEKTROSVIAZ No. 3, March 1959, pp 73-81, Complete Paper translated in AUTOMATION EXPRESS, VOL. 1, April 1959, p 19.
59. Kodis, R. D., Ruhman, S. and Woo, W. D., "Magnetic Shift Register Using One Core Per Bit", I.R.E. CONVENTION RECORD, Part 7, 1953, pp 38-42.
60. Ky Fan, "On Systems of Linear Inequalities", ANNALS OF MATHEMATICS STUDIES, No. 38, edited by H. W. Kuhn and A. W. Tucker, Princeton University Press, 1956.

61. Lincoln, Andrew J., U. S. Ballistic Research Laboratories, "Ferromagnetic Core Logical Circuitry and its Application to Digital Computers", BRL REPORT 9111, August 1955.
62. Lockhart, N. F., "Logic by Ordered Flux Changes in Multipath Ferrite Cores", I.R.E. NATIONAL CONVENTION RECORD, 1958, Part 4, pp 268-278.
63. Loev, D., Miehle, W., Paivinen, J., and Wylen, J., "Magnetic Core Circuits for Digital Data Processing System", PROCEEDINGS OF THE I.R.E., February 1956, pp 154-162.
64. McNaughton, R., "Unate Truth Function", I.R.E. TRANSACTIONS ON ELECTRONIC COMPUTERS, VOL. EC-10, No. 1, March 1961.
65. Minnick, Robert C., LINEAR INPUT LOGICAL CIRCUITRY, Burroughs Corporation, ElectroData Division, October 22, 1958, Engineering Technical Memorandum, ETM Number 39.
66. Minnick, Robert C., "Synthesis of Linear-Input Logic by the Simplex Method", delivered at the Sixth Annual Symposium on Computers and Data Processing of the Denver Research Institute, July 30, 1959.
67. Morgan, W. L., "Bibliography of Digital Magnetic Circuits and Materials", I.R.E. TRANSACTIONS ON ELECTRONIC COMPUTERS, VOL. EC-8, No. 2, June 1959, pp 148-58.
68. Myers, A. S. Jr., "Self-Propagating Care of Logic", PROCEEDINGS OF NONLINEAR MAGNETIC AND MAGNETIC AMPLIFIER CONFERENCE, October 1960, p 74.
69. Marshall, T. G. Jr., and Andrews, L. J., National Cash Register Company, "Research on Electro-Optical and Magnetic Core Logic", REPORT for 15 September 1956 - 15 September 1957 on AUTOMATIC COMPUTATION TECHNIQUES AND COMPONENTS, p 43, (WADC Technical Report 57-607) (Contract AF 33(616)3837), Unclassified report, Part 1 - Phosphor-hot-conductor Research, Part 2 - Core Logic Summary.
70. National Cash Register Company, "Core Computer Group, Interim Research Progress Report No. 4", 1 April - 1 August 1957, (Bound with National Cash Register Co., Dayton, Ohio, Contract AF 33(616)3837, INTERIM RESEARCH PROGRESS REPORT No. 4 AD-142 984), Unclassified report.

71. Newell, A. F., "The Design of Logical Circuits Using Transistors and Square Loop Ferrite Cores", MULLARD TECHNICAL COMMUNICATIONS, VOL. 4, October 1958, pp 110-20.
72. Newhouse, V. L. and Prywess, N. S., "High Speed Shift Registers Using One Core per Bit", I.R.E. TRANSACTIONS OF THE PGEC, September 1956, pp 114-120.
73. Newhouse, V. L., "The Utilization of Domain Wall Viscosity in Data-handling Devices", I.R.E. PROCEEDINGS, VOL. 45, November 1957, pp 1484-1492.
74. Newhouse, V. L., "A Review of Magnetic and Ferroelectric Computing Components", ELECTRONIC ENGINEERING, VOL. 26, May 1954, pp 192-9.
75. Oakland, Lewis J. and Rossing, Thomas D., "Coincident-current Nondestructive Readout from Thin Magnetic Films, JOURNAL OF APPLIED PHYSICS, Supplement to VOL. 30, April 1959, pp 54S-55S.
76. Olsen, Kenneth H., "A Magnetic Matrix Switch and its Incorporation into a Coincident-current Memory", Cambridge, June 6, 1952, p 96, (Report R-211).
77. Paull, M. C. and McCluskey, E. J. Jr., "Boolean Functions Realizable with Single Threshold Devices", PROCEEDINGS, I.R.E., VOL. 48, July 1960, pp 1335-1337.
78. Rajchman, J. A., "Magnetic Switching", WESTERN JOINT COMPUTER CONFERENCE (see Abstr. 4255, 1959), pp 107-16.
79. Rajchman, J. A. and Lo, A. W., "The Transfluxor", PROCEEDINGS OF THE I.R.E., VOL. 43, March 1956, pp 321-338, RCA REVIEW, VOL. 16, June 1955, pp 303-311.
80. Rajchman, J. A., "Magnetics for Components", REC REVIEW, March 1959.
81. Ramey, R. A., "The Single-Core Magnetic Amplifier as a Component Element", A.I.E.E. TRANSACTIONS, Part I, VOL. 71, 1952, pp 442-6.
82. Russell, L. A., "Diodeless Magnetic Core Logical Circuits", I.R.E. CONVENTION RECORD, Part 4, 1957, pp 106-114.

83. Sands, Eugene A., Magnetics Research Co., Chappaqua, N. Y., "Magnetic Core Switches as Logical Elements in Computers", May 1954, VOL. 1, including illustrations, (WADC Technical Report No. 54-232), (Contract AF 33(600)24923), Unclassified.
84. Sands, Eugene A., "Magnetic Core Switches as Logical Elements in Computers", I.R.E CONVENTION RECORD, 1953, Pt. 7, p 37, (Abstract).
85. Scarrott, G. G., and Others, "The Design and Use of Logical Devices Using Saturable Magnetic Cores", INSTITUTION OF ELECTRICAL ENGINEERS, PROCEEDINGS, VOL. 103B, 1956, Supplement 1-3, pp 302-312, (Proceedings of Convention on Digital Computer Techniques).
86. Schultz, Carl J., Massachusetts Institute of Technology, Digital Computer Laboratory, "Magnetic Core Matrix Switch Adder", Cambridge, March 9, 1953, p 6, (Memorandum M-1883).
87. Stabler, E. P., General Electric, "Square Loop Magnetic Logic Circuits", To be published in PROCEEDINGS 1959 WESTERN JOINT COMPUTER CONFERENCE, San Francisco, March 3-5, 1959, (Publication date - June 1959).
88. Stram, O. and Einhorn, S., "Magnetic Logical Transducers", TR 59-54, Burroughs Corporation, Research Center, Paoli, Pa., September 30, 1959.
89. Stram, O. and Einhorn, S., "Magnetic Logical Transducers", Technical Report TR 59-68, Burroughs Corporation, December 21, 1959.
90. Tancrell, Roger H., "Impulsive Selection for Core Logic", JOURNAL OF APPLIED PHYSICS, Supplement to VOL. 32, No. 3, March 1961, pp 40S.
91. Terada, Hiroshi, "The Parametron - An Amplifying Logic Element", CONTROL ENGINEERING, VOL. 6, April 1959, pp 110-115.
92. VanNice, R. I., "Magnetic Logic Circuit Control System Design Considerations", A.I.E.E. TRANSACTIONS, VOL. 75, Part 1, 1956, pp 595-600.
93. Wang, An, "Magnetic Delay Line Storage", PROCEEDINGS OF THE I.R.E., VOL. 39, April 1959, pp 401-407.

94. Wang, An, "Static Magnetic Storage and Delay Lines", JOURNAL OF APPLIED PHYSICS, VOL. 31, January 1950, pp 49-54.
95. Winder, R. O., "Single-Stage Threshold Logic", NEC-AIEE, Joint Symposium on Switching Circuit Theory and Logical Design, Chicago, Illinois, October 11-12, 1960.
96. Woo, W. D., "Magnetic Core Logical Circuits", International Symposium on Switching Theory, Harvard University, Cambridge, Massachusetts, April 2-5, 1957.

GLOSSARY

- H_1 The applied magnetic field, or magnetizing force. $H_1 = NI/\ell$ in M. K. S. unit.
- H_0 The unit of applied magnetic field. $H_1 = M H_0$ where M take positive and negative integer values and zero. $H_0 > H_c$, the coercive force, and depends on the desired speed of switching.
- I The amplitude of the applied constant current pulse taken so that for $N = 1$, $NI/\ell = I/\ell = H_0$.
- C A matrix operator which applied to the solution vector transforms it in accordance with a given set of symmetry operations on the inputs.
- P A permutation matrix which transforms a function in accordance with a set of symmetry operations on the inputs to the switching circuit generating the function.
- T_t The transpose of T, the truth table with an added column of "ones" considered as a matrix.
- S A column vector representing the solution, or design, of a switching circuit. The terms $q, N_i, 0 \leq i \leq n$ are the input values (weights) of the drivers, x_i .
- $F_A, G_A,$ etc. The partitions of a function, A , which is not realizable in a single core.

- x_i The input variables for $1 \leq i \leq n$. For $n = 0$, x_0 represents the bias which is always present.
- f_r For $0 \leq r \leq 2^n - 1$, the truth value of the r th state of the truth function.
- M_r The number of units of H_0 for the r th state of the function. M_r is the sum of the input values (weights) of the drivers which are "on" in the r th state.
- δ_S A number which arises in testing whether a hypothetical profile is consistent. If $\delta_S \geq 2$, the profile is not consistent.